ANALOG DEVICES

DSP Microcomputer

Preliminary Technical Data ADSP-2192

ADSP-2192 DUAL-CORE DSP FEATURES

320 MIP Dual ADSP-219x DSP in a 144-lead LQFP package with PCI, USB, Sub-ISA, and CardBus Interfaces

3.3V/5V PCI 2.2 Compliant 33MHz / 32-bit Interface with Bus Mastering over four DMA Channels with Scatter-Gather Support

Integrated USB 1.1 Compliant Interface

AC '97 serial interface supports external modem, handset, and audio codecs

Dual 160 MIPS ADSP-219x DSPs with 140K Words of Memory and 4K x 16-bit Shared Data Memory

DSP P0 Memory Includes: 64K x 16-bit Data Memory, 16K x 24-bit Program Memory, and Boot ROM

DSP P1 Memory Includes: 32K x 16-bit Data Memory, 16K x 24-bit Program Memory, and Boot ROM

ADSP-219X DSP CORE FEATURES

6.25 ns Instruction Cycle Time (Internal), for up to 160 MIPS Sustained Performance

ADSP-218x Family Code Compatible with the Same Easy to Use Algebraic Syntax

Single-cycle Instruction Execution

Dual Purpose Program Memory for Both Instruction and Data Storage

Fully Transparent Instruction Cache Allows Dual Operand Fetches in Every Instruction Cycle

Unified Memory Space Permits Flexible Address Generation, Using Two Independent DAG Units

Independent ALU, Multiplier/Accumulator, and Barrel Shifter Computational Units with Dual 40-bit Accumulators

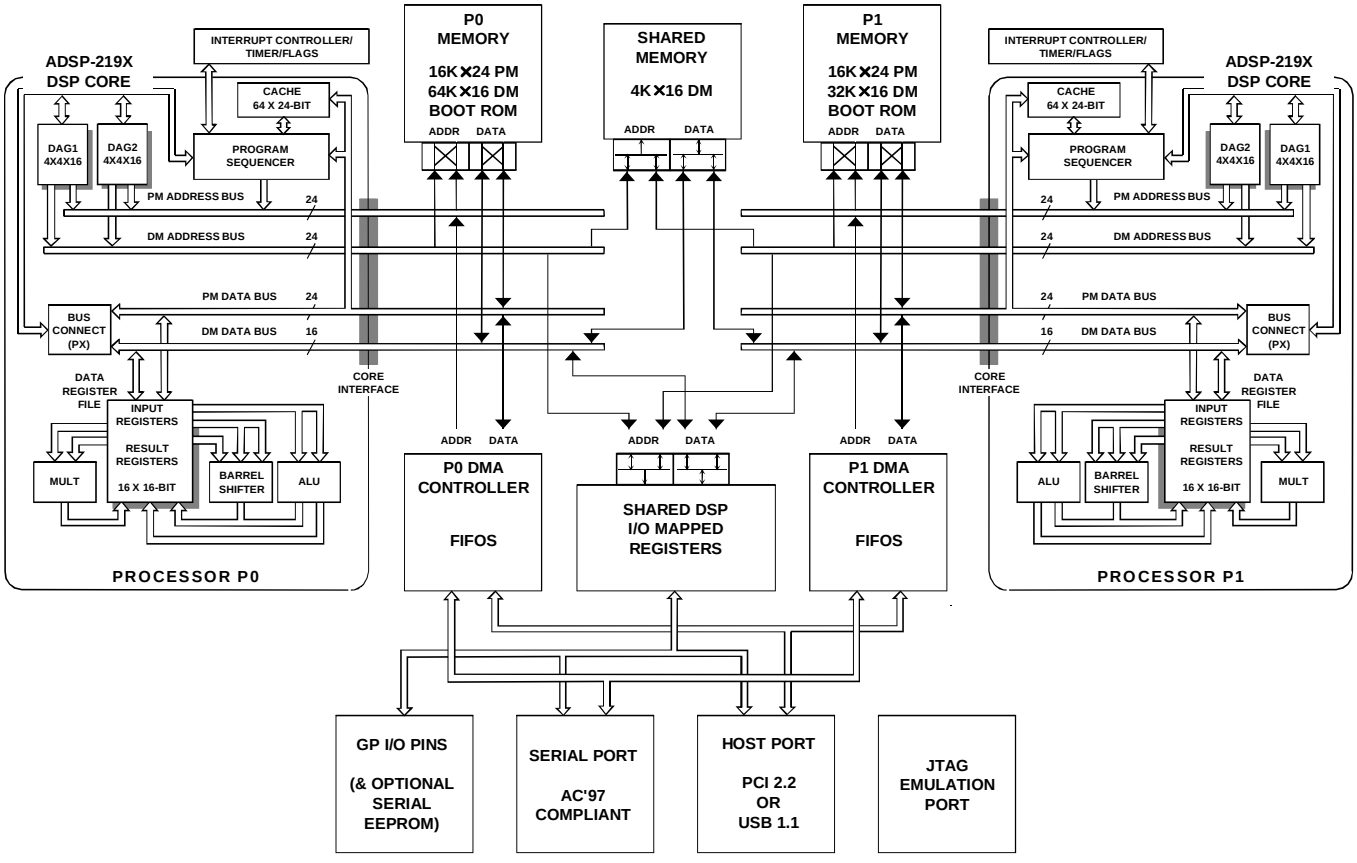

Figure 1. ADSP-2192 Dual-Core DSP Block Diagram

REV. PrA

One Technology Way, P.O.Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel:781/329-4700 World Wide Web Site: http://www.analog.com
Fax:781/326-8703

ADSP-219X DSP CORE FEATURES (CONTINUED)

- Single-Cycle Context Switch Between Two Sets of Computational and DAG Registers
- Parallel Execution of Computation and Memory Instructions
- Pipelined Architecture Supports Efficient Code Execution at Speeds up to 160 MIPS
- Register File Computations with All Non-conditional, Non-parallel Computational Instructions
- Powerful Program Sequencer Provides Zero- Overhead Looping and Conditional Instruction Execution
- Architectural Enhancements for Compiled C/C++ Code Efficiency
- Architecture Enhancements Beyond ADSP-218x Family are Supported with Instruction Set Extensions for Added Registers, Ports, and Peripherals

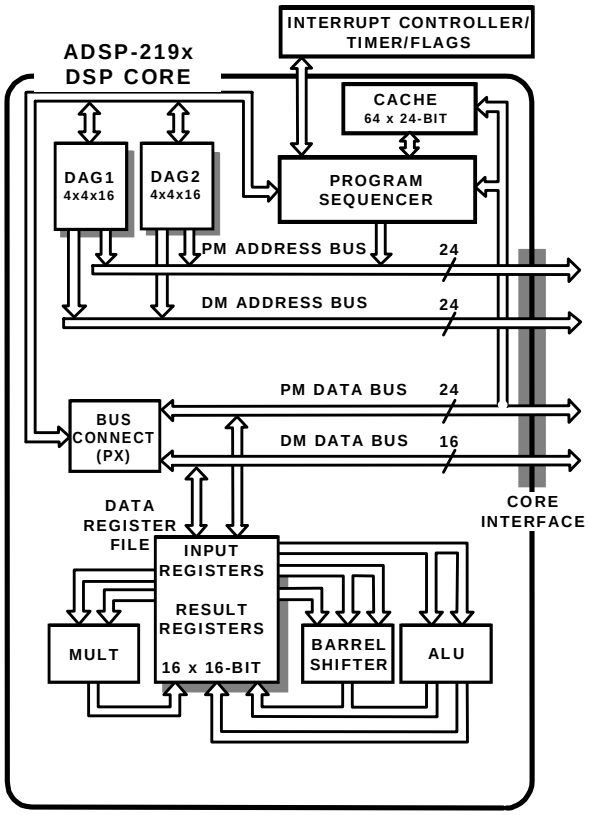

Figure 2. ADSP-219x DSP Core

ADSP-2192 DSP FEATURES (CONTINUED)

- Two ADSP-219x core processors (P0 and P1) on each ADSP-2192 DSP chip
- 80K words of on-chip RAM on P0, configured as 64K words on-chip 16-bit RAM for Data Memory and 16K words on-chip 24-bit RAM for Program Memory
- 48K words of on-chip RAM on P1, configured as 32K words on-chip 16-bit RAM for Data Memory and 16K words on-chip 24-bit RAM for Program Memory
- 4K words of additional on-chip RAM shared by both cores, configured as 4K words on-chip 16-bit RAM
- Flexible power management with selectable power-down and idle modes
- Programmable PLL supports frequency multiplication, enabling full-speed operation from low-speed input clocks
- 2.5V internal operation supports 3.3V/5.0V compliant I/O
- A Host port that supports either PCI (PCI interface and CardBus) or USB (USB 1.1 compliant) interfaces; both with DMA capability
- Sub-ISA Interface
- An AC'97 port supporting AC'97 Revision 2.1 compliant interface for External Audio, Modem, and Handset Codecs with DMA capability
- Eight dedicated general-purpose I/O pins with integrated interrupt support
- Each DSP core has a programmable 32-bit interval timer
- Five DMA channels available on each core
- Boot methods include booting through PCI port, USB port, or serial EEPROM
- JTAG Test Access Port supports on-chip emulation and system debugging
- 144-lead LQFP package (20x20x1.4mm)

General note

This data sheet provides preliminary information for the ADSP-2192 Digital Signal Processor.

GENERAL DESCRIPTION

The ADSP-2192 is a single-chip microcomputer optimized for digital signal processing (DSP) and other high speed numeric processing applications.

The ADSP-2192 combines the ADSP-219x family base architecture (three computational units, two data address generators and a program sequencer) into a chip with two core processors. The ADSP-2192 includes a PCI-compatible port, a USB-compatible port, an AC'97-compatible port, a DMA controller, a programmable timer, general purpose Programmable Flag pins, extensive interrupt capabilities, and on-chip program and data memory spaces.

The ADSP-2192 architecture is code compatible with ADSP-218x family DSPs. Though the architectures are compatible, the ADSP-2192 architecture has many enhancements over the ADSP-218x architecture. The enhancements to computational units, data address genera-

tors, and program sequencer make the ADSP-2192 more flexible and even easier to program than the ADSP-218x DSPs.

Indirect addressing options provide addressing flexibility—premodify with no update, post-modify with update, pre- and post-modify by an immediate 8-bit, two's-complement value and base address registers for easier implementation of circular buffering.

The ADSP-2192 integrates 64K words of on-chip memory configured as 32K words (24-bit) of program RAM, and 96K words (16-bit) of data RAM. Power-down circuitry is also provided to meet the low power needs of battery operated portable equipment. The ADSP-2192 is available in a 144-lead LQFP package.

Fabricated in a high speed, low power, CMOS process, the ADSP-2192 operates with a 6.25-ns instruction cycle time (160 MIPS). All instructions, except two multiword instructions, can execute in a single DSP cycle.

The ADSP-2192's flexible architecture and comprehensive instruction set support multiple operations in parallel. For example, in one processor cycle, each DSP core within the ADSP-2192 can:

- Generate an address for the next instruction fetch
- Fetch the next instruction
- Perform one or two data moves
- Update one or two data address pointers
- Perform a computational operation

These operations take place while the processor continues to:

- Receive and/or transmit data through the Host port (PCI or USB interfaces)
- Receive or transmit data through the AC'97
- Decrement the two timers

DSP Core Architecture

The ADSP-2192 instruction set provides flexible data moves and multifunction (one or two data moves with a computation) instructions. Every single-word instruction can be executed in a single processor cycle. The ADSP-2192 assembly language uses an algebraic syntax for ease of coding and readability. A comprehensive set of development tools supports program development.

Figure 1 on page 1 shows the architecture of the ADSP-219x dual-core DSP. Each core contains three independent computational units: the ALU, the multiplier/accumulator (MAC) and the shifter. The computational units process 16-bit data from the register file and have provisions to support multiprecision computations. The ALU performs a standard set of arithmetic and logic operations; division primitives are also supported. The MAC performs single-cycle multiply, multiply/add and multiply/subtract operations. The MAC has two 40-bit accumulators, which help with overflow. The shifter performs logical and arithmetic shifts, normalization, denormalization, and derive exponent operations. The shifter can be used to efficiently implement numeric format control, including multiword and block floating-point representations.

Register-usage rules influence placement of input and results within the computational units. For most operations, the computational units' data registers act as a data register file, permitting any input or result register to provide input to any unit for a computation. For feedback operations, the computational units let the output (result) of any unit be input to any unit on the next cycle. For conditional or multifunction instructions, there are restrictions on which data registers may provide inputs or receive results from each computational unit. For more information, see the *ADSP-219x DSP Instruction Set Reference*.

A powerful program sequencer controls the flow of instruction execution. The sequencer supports conditional jumps, subroutine calls, and low interrupt overhead. With internal loop counters and loop stacks, the ADSP-2192 executes looped code with zero overhead; no explicit jump instructions are required to maintain loops.

Two data address generators (DAGs) provide addresses for simultaneous dual operand fetches. Each DAG maintains and updates four 16-bit address pointers. Whenever the pointer is used to access data (indirect addressing), it is pre- or post-modified by the value of one of four possible modify registers. A length value and base address may be associated with each pointer to implement automatic modulo addressing for circular buffers. Page registers in the DAGs allow linear or circular addressing within 64 Kword boundaries of each of the memory pages, but these buffers may not cross page boundaries. Secondary registers duplicate all the primary registers in the DAGs; switching between primary and secondary registers provides a fast context switch.

Efficient data transfer in the core is achieved with the use of internal buses:

- Program Memory Address (PMA) Bus
- Program Memory Data (PMD) Bus
- Data Memory Address (DMA) Bus
- Data Memory Data (DMD) Bus

Program memory can store both instructions and data, permitting the ADSP-2192 to fetch two operands in a single cycle, one from program memory and one from data memory. The DSP's dual memory buses also let the ADSP-2192 core fetch an operand from data memory and the next instruction from program memory in a single cycle.

DSP Peripherals Architecture

Figure 1 on page 1 shows the DSP's on-chip peripherals, which include the Host port (PCI or USB), AC'97 port, JTAG test and emulation port, flags, and interrupt controller.

The ADSP-2192 can respond to up to thirteen interrupts at any given time. A list of these interrupts appears in Table 1.

The AC'97 Codec port on the ADSP-2192 provides a complete synchronous, full-duplex serial interface. This interface completely supports the AC'97 standard.

The ADSP-2192 provides up to eight general-purpose I/O pins, which are programmable as either inputs or outputs. These pins are dedicated general purpose Programmable Flag pins.

The programmable interval timer generates periodic interrupts. A 16-bit count register (TCOUNT) is decremented every n cycles where n-1 is a scaling value stored in a 16-bit register (TSCALE). When the value of the count register reaches zero, an interrupt is generated and the count register is reloaded from a 16-bit period register (TPERIOD).

Memory Architecture

The ADSP-2192 provides 140K words of on-chip SRAM memory. This memory is divided into Program and Data Memory blocks in each DSP's memory map. In addition to the internal memory space, the two cores can address two additional and separate off-core memory spaces: I/O space and shared memory space, as shown in Figure 3 on page 4.

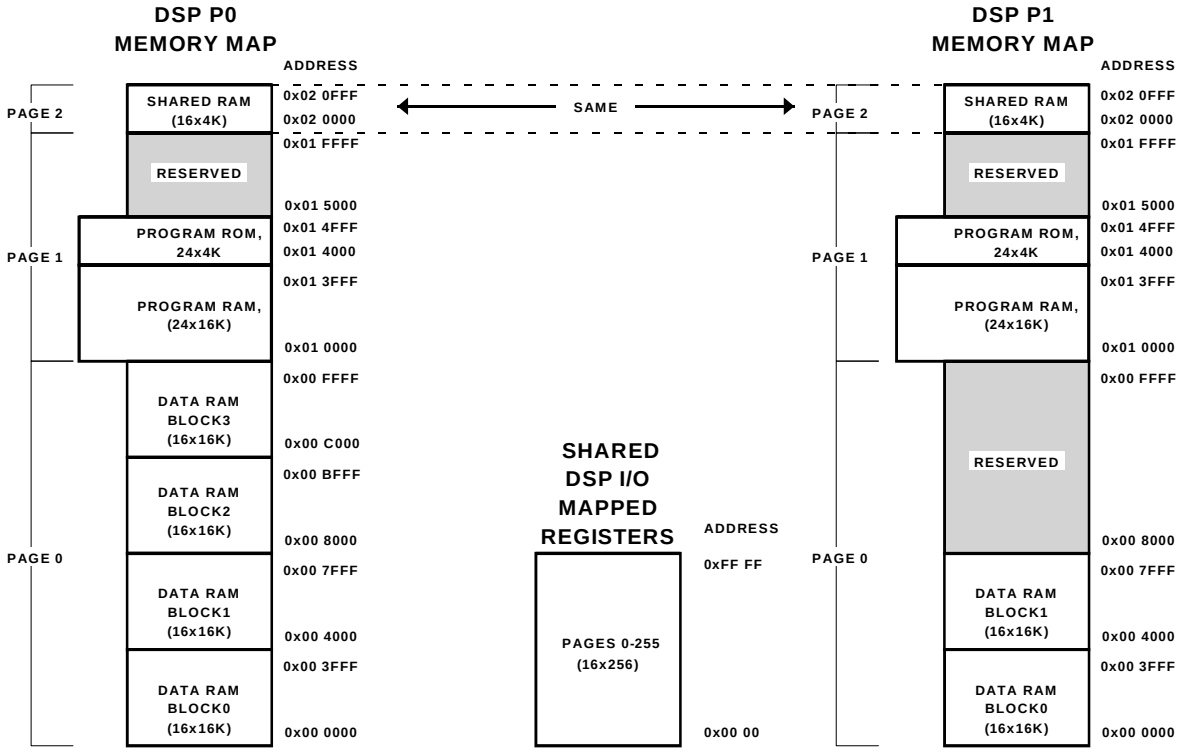

Figure 3. ADSP-2192 Internal/External Memory, Boot Memory, and I/O Memory Maps

The ADSP-2192's two cores can access 80K and 48K locations that are accessible through two 24-bit address buses, the PMA and DMA buses. The DSP uses slightly different

mechanisms to generate a 24-bit address for each bus. The DSP has three functions that support access to the full memory map.

- The DAGs generate 24-bit addresses for data fetches from the entire DSP memory address range. Because DAG index (address) registers are 16 bits wide and hold the lower 16-bits of the address, each of the DAGs has its own 8-bit page register (DMPGx) to hold the most significant eight address bits. Before a DAG generates an address, the program must set the DAG's DMPGx register to the appropriate memory page.
- The Program Sequencer generates the addresses for instruction fetches. For relative addressing instructions, the program sequencer bases addresses for relative jumps, calls, and loops on the 24-bit Program Counter (PC). In direct addressing instructions (two-word instructions), the instruction provides an immediate 24-bit address value. The PC allows linear addressing of the full 24 bit address range.
- For indirect jumps and calls that use a 16-bit DAG address register for part of the branch address, the Program Sequencer relies on an 8-bit Indirect Jump page (IJPG) register to supply the most significant eight address bits. Before a cross page jump or call, the program must set the program sequencer's IJPG register to the appropriate memory page.

Each ADSP-219x DSP core has an on-chip ROM that holds boot routines. For more information, see "Booting Modes" on page 31.

Interrupts

The interrupt controller lets the DSP respond to thirteen interrupts with minimum overhead. The controller implements an interrupt priority scheme as shown in Table 1 on page 5. Applications can use the unassigned slots for software and peripheral interrupts. The DSP's Interrupt Control (ICNTL) register (shown in Table 3 on page 6) provides controls for global interrupt enable, stack interrupt configuration, and interrupt nesting.

Table 2 on page 5 shows the interrupt vector and DSP-to-DSP semaphores at reset of each of the peripheral interrupts. The peripheral interrupt's position in the IMASK and IRPTL register and its vector address depend on its priority level, as shown in Table 1 on page 5.

Table 1. Interrupt Vector Table

Bit	Priority	Interrupt	Vector Address Offset[1]
0	1	Reset (non-maskable)	0x00
1	2	Powerdown (non-maskable)	0x04
2	3	Kernel interrupt (single step)	0x08
3	4	Stack Status	0x0C
4	5	Mailbox	0x10
5	6	Timer	0x14
6	7	Reserved	0x18
7	8	PCI Bus Master	0x1C
8	9	DSP-DSP	0x20
9	10	FIFO0 Transmit	0x24
10	11	FIFO0 Receive	0x28
11	12	FIFO1 Transmit	0x2C
12	13	FIFO1 Receive	0x30
13	14	Reserved	0x34
14	15	Reserved	0x38
15	16	AC'97 Frame	0x3C

[1] The interrupt vector address values are represented as offsets from address 0x01 0000. This address corresponds to the start of Program Memory in DSP P0 and P1.

Table 2. DSP-to-DSP Semaphores Register Table

Flag Bit	Direction	Function	DSP Core Flag In
0	Output	DSP-DSP Semaphore 0	
1	Output	DSP-DSP Semaphore 1	
2	Output	DSP-DSP Interrupt	

Table 2. DSP-to-DSP Semaphores Register Table

Flag Bit	Direct-ion	Function	DSP Core Flag In
3		Reserved	
4		Reserved	
5		Reserved	
6		Reserved	
7	Output	Register Bus Lock	
8	Input	DSP-DSP Semaphore 0	0
9	Input	DSP-DSP Semaphore 1	1
10	Input	DSP-DSP Interrupt	2
11	Input	Reserved	
12	Input	AC'97 Register - PDC Bus Access Status	4
13	Input	PDC Interface Busy Status (write from DSP pending)	5
14	Input	Reserved	
15	Input	Register Bus Lock Status	7

Interrupt routines can either be nested with higher priority interrupts taking precedence or processed sequentially. Interrupts can be masked or unmasked with the IMASK register. Individual interrupt requests are logically ANDed with the bits in IMASK; the highest priority unmasked interrupt is then selected. The emulation, power down, and reset interrupts are nonmaskable with the IMASK register, but software can use the DIS INT instruction to mask the power down interrupt.

Table 3. Interrupt Control (ICNTL) register bits

Bit	Description
0–3	Reserved
4	Interrupt nesting enable
5	Global interrupt enable
6	Reserved
7	MAC biased rounding enable
8–9	Reserved
10	PC stack interrupt enable

Table 3. Interrupt Control (ICNTL) register bits

Bit	Description
11	Loop stack interrupt enable
12	Low power idle enable
13–15	Reserved

The IRPTL register is used to force and clear interrupts. On-chip stacks preserve the processor status and are automatically maintained during interrupt handling. To support interrupt, loop, and subroutine nesting, the PC stack is 33-levels deep, the loop stack is eight-levels deep, and the status stack is sixteen-levels deep. To prevent stack overflow, the PC stack can generate a stack level interrupt if the PC stack falls below 3 locations full or rises above 28 locations full.

The following instructions globally enable or disable interrupt servicing, regardless of the state of IMASK.

```
ENA INT;
DIS INT;
```

At reset, interrupt servicing is disabled.

For quick servicing of interrupts, a secondary set of DAG and computational registers exist. Switching between the primary and secondary registers lets programs quickly service interrupts, while preserving the DSP's state.

DMA Controller

The ADSP-2192 has a DMA controller that supports automated data transfers with minimal overhead for the DSP core. Cycle stealing DMA transfers can occur between the ADSP-2192's internal memory and any of its DMA capable peripherals. Additionally, DMA transfers can also be accomplished between any of the DMA capable peripherals. DMA capable peripherals include the PCI and AC'97 ports. Each individual DMA capable peripheral has a dedicated DMA channel. DMA sequences do not contend for bus access with the DSP core; instead, DMAs "steal" cycles to access memory.

All DMA transfers use the Program Memory (PMA/PMD) buses shown in Figure 1 on page 1.

External Interfaces

There are several different interfaces supported on the ADSP-2192. These include both internal and external interfaces. The three separate PCI configuration spaces are programmable to set up the device in various Plug-and-Play configurations.

The ADSP-2192 provides the following types of external interfaces: PCI, USB, Sub-ISA, CardBus, AC'97, and serial EEPROM. The following sections discuss those interfaces.

PCI 2.2 Host Interface

The ADSP-2192 includes a 33MHz, 32 bit bus master PCI interface that is compliant with revision 2.2 of the PCI specification. This interface supports the high data rates.

USB 1.1 Host Interface.

The ADSP-2192 USB interface enables the host system to configure and attach a single device with multiple interfaces and various endpoint configurations. The advantages of this design include:

- Programmable descriptors and class-specific command interpreter.
- An on-chip 8052 compatible MCU allows the user to soft download different configurations and support standard or class-specific commands.
- Total of 8 user-defined endpoints provided. Endpoints can be configured as either BULK, ISO, or INT and can be grouped and assigned to any interface.

Sub-ISA Interface

In systems which combine the ADSP-2192 chip with other devices on a single PCI interface, the ADSP-2192 Sub-ISA mode is used to provide a simpler interface which bypasses the ADSP-2192's PCI interface. In this mode the Combo Master assumes all responsibility for interfacing the function to the PCI bus, including provision of Configuration Space registers for the ADSP-2192 system as a separate PnP function. In Sub-ISA Mode the PCI Pins are reconfigured for ISA operation.

CardBus Interface

The CardBus standard provides higher levels of performance than the 16-bit PC Card standard. For example, 32-bit CardBus cards are able to take advantage of internal bus speeds that can be as much as four- to six-times faster than 16-bit PC Cards. This design provides for a compact, rugged card that can be inserted completely within its host computer without any external cabling.

Since CardBus performance attains the same high level as the host platform's internal (PCI) system bus, it is an excellent way to add high speed communications to the notebook form factor. In addition, CardBus PC Cards operate at a power-saving 3.3 volts, extending battery life in most configurations.

This new 32-bit CardBus technology provides up to 132Mbytes per second of bandwidth. This performance makes CardBus an ideal vehicle to furnish the demands of high throughput communications such as ADSL.

CardBus PC Cards generate less heat and consume less power. This is attained by:

- Low voltage operation at 3.3V.
- Software control of clock speed.
- Advanced power management mechanism

AC'97 2.1 External Codec Interface

The industry standard AC'97 serial interface (AC-Link) incorporates a 7-pin digital serial interface that links compliant codecs to the ADSP-2192. The ACLink implements a bi-directional, fixed rate, serial PCM digital stream. It handles multiple input and output audio streams as well as control and status register accesses using a time division multiplex scheme.

Serial EEPROM Interface

The Serial EEPROM for the ADSP-2192 can overwrite the following information which is returned during the USB GET DEVICE DESCRIPTOR command. During the Serial EEPROM initialization procedure, the DSP is responsible for writing the USB Descriptor Vendor ID, USB Descriptor Product ID, USB Descriptor Release Number, and USB Descriptor Device Attributes registers to change the default settings.

All descriptors can be changed when downloading the RAM-based MCU re-numeration code, except for the Manufacturer and Product, which are supported in the CONFIG DEVICE and cannot be overwritten or changed by the Serial EEPROM.

- Vendor ID (0x0456 ADI)
- Product ID (0x2192)
- Device Release Number (0x0100)
- Device Attributes (0x80FA): SP (1=self-powered, 0=bus-powered, default=0); RW (1=have remote wake-up capability, 0=no remote wake-up capability, deafult=0); C[7:0] (power consumption from bus expressed in 2mA units; default = 0xFA 500mA)
- Manufacturer (ADI)
- Product (ADI Device)

Internal Interfaces

The ADSP-2192 provides three types of internal interfaces: registers, codec, and DSP memory buses. The following sections discuss those interfaces.

Register Interface

The register interface allows the PCI interface, USB interface, and both DSPs to communicate with the I/O Registers. These registers map into DSP, PCI, and USB I/O spaces.

Register Spaces

Several different register spaces are defined on the ADSP-2192, as described in the following sections.

PCI Configuration Space

These registers control the configuration of the PCI Interface. Most of these registers are only accessible via the PCI Bus although a subset is accessible to the DSP for configuration during the boot.

DSP Core Register Space

Each DSP has an internal register that is accessible with no latency. These registers are accessible only from within the DSP, using the REG( ) instruction.

Peripheral Device Control Register Space

This Register Space is accessible by both DSPs, the PCI, Sub-ISA, and USB Buses. Note that certain sections of this space are exclusive to either the PCI, USB, or Sub-ISA Buses. These registers control the operation of the peripherals of the ADSP-2192. The DSP accesses these registers using the IO( ) instruction.

USB Register Space

These registers control the operation and configuration of the USB Interface. Most of these registers are only accessible via the USB Bus, although a subset is accessible to the DSP.

Card Bus interface

The ADSP-2192's PC Card Bus interface meets the state and timing specifications defined for PCMCIA's PC Card Bus Standard April 1998 Release 6.1. It supports up to three card functions. Multiple function PC cards require a separate set of Configuration registers per function. A primary Card Information Structure common to all functions is required. Separate secondary Card Information Structures, one per function, are also required. Data for each CIS is loaded by the DSP during bootstrap loading.

The host PC can read the CIS data at any time. If needed, the WAIT control can be activated to extend the read operation to meet bus write access to the CIS data.

Using the PCI Interface

The ADSP-2192 includes a 33-Mhz, 32-bit PCI interface to provide control and data paths between the part and the host CPU. The PCI interface is compliant with the PCI Local Bus Specification Revision 2.2. The interface supports both bus mastering as well as bus target interfaces. The PCI Bus Power Management Interface Specification Revision 1.1 is supported and additional features as needed by PCI designs are included.

Target/Slave Interface

The ADSP-2192 PCI interface contains three separate functions, each with its own configuration space. Each function contains four base address registers used to access ADSP-2192 control registers and DSP memory. Base Address Register (BAR) 1 is used to point to the control registers. The addresses specified in these tables are offsets from BAR1 in each of the functions. PCI memory-type accesses are used to read and write the registers.

DSP memory accesses use BAR2 or BAR3 of each function. BAR2 is used to access 24-bit DSP memory; BAR3 accesses 16-bit DSP memory. Maps of the BAR2 and BAR3 registers appear in Table 8 on page 14 and Table 9 on page 15.

The lower half of the allocated space pointed to by each DSP memory BAR is the DSP memory for DSP core P0. The upper half is the memory space associated with DSP core P1. PCI transactions to and from DSP memory use the DMA function within the DSP core. Thus each word transferred to or from PCI space uses a single DSP clock cycle to perform the internal DSP data transfer. Byte-wide accesses to DSP memory are not supported.

I/O type accesses are supported via BAR4. Both the control registers accessible via BAR1 and the DSP memory accessible via BAR2 and BAR3 can be accessed with I/O accesses. Indirect access is used to read and write both the control registers and the DSP memory. For the control register accesses, a address register points to the word to be accessed while a separate register is used to transfer the data. Read/write control is part of the address register. Only 16-bit accesses are possible via the I/O space.

A separate set of registers is used to perform the same function for DSP memory access. Control for these accesses includes a 24-bit/16-bit select as well as direction control. The data register for DSP memory accesses is a full 24-bits wide. 16-bit accesses will be loaded into the lower 16-bits of the register. Table 10 on page 17 lists the registers directly accessible from BAR4.

Bus Master Interface

As a bus master, the PCI interface can transfer DMA data between system memory and the DSP. The control registers for these transfers are available both to the host and to the DSPs. Four channels of bus-mastering DMA are supported on the ADSP-2192.

Two channels are associated with the receive data and two are associated with the transmit data. The internal DSPs will typically control initiation of bus master transactions. DMA host bus master transfers can specify either standard circular buffers in system memory or perform scatter-gather DMA to host memory.

Each bus master DMA channel includes 4 registers to specify a standard circular buffer in system memory. The Base Address points to the start of the circular buffer. The Current Address is a pointer to the current position within that buffer. The Base Count specifies the size of the buffer in bytes, while the Current Count keeps track of how many bytes need to be transferred before the end of the buffer is reached. When the end of the buffer is reached, the channel can be programmed to loop back to the beginning and continue the transfers. When this looping occurs, a Status bit will be set in the DMA Control Register.

When transferring samples to and from DSP memory, the PCI DMA controller can be programmed to perform scatter-gather DMA. This mode allows the data to be split up in memory, and yet be able to be transferred to and from the ADSP-2192 without processor intervention. In scatter-gather mode, the DMA controller can read the memory address and word count from an array of buffer descriptors called the Scatter-Gather Descriptor (SGD) table. This allows the DMA engine to sustain DMA transfers until all buffers in the SGD table are transferred.

To initiate a scatter-gather transfer between memory and the ADSP-2192, the following steps are involved:

1. Software driver prepares a SGD table in system memory. Each descriptor is eight bytes long and consists of an address pointer to the starting address and the transfer count of the memory buffer to be transferred. In any given SGD table, two consecutive SGDs are offset by eight bytes and are aligned on a 4-byte boundary. Each SGD contains:
   a. Memory Address (Buffer Start) – 4 bytes
   b. Byte Count (Buffer Size) – 3 bytes
   c. End of Linked List (EOL) – 1 bit (MSBit)
   d. Flag – 1 bit (MSBit – 1)
2. Initialize DMA control registers with transfer specific information such as number of total bytes to transfer, direction of transfer, etc.
3. Software driver initializes the hardware pointer to the SGD table.
4. Engage scatter-gather DMA by writing the start value to the PCI channel Control/Status register.
5. The ADSP-2192 will then pull in samples as pointed to by the descriptors as needed by the DMA engine. When the EOL is reached, a status bit will be set and the DMA will end if the data buffer is not to be looped. If looping is to occur, DMA transfers will continue from the beginning of the table until the channel is turned off.
6. Bits in the PCI Control/Status register control whether an interrupt occurs when the EOL is reached or when the FLAG bit is set.

Scatter-gather DMA uses four registers. In scatter-gather mode the functions of the registers are mapped as follows:

Table 4. Register-Mapping in Scatter-Gather Mode

Standard Circular Buffer Mode	Scatter-Gather Mode Function
Base Address	SGD Table Pointer
Current Address	SGD Current Pointer Address
Base Count	SGD Pointer
Current Count	Current SGD Count

In either mode of operation, interrupts can be generated based upon the total number of bytes transferred. Each channel has two 24-bit registers to count the bytes transferred and generate interrupts as appropriate. The Interrupt Base Count register specifies the number of bytes to transfer prior to generating an interrupt. The Interrupt Count register specifies the current number left prior to generating the interrupt. When the Interrupt Count register reaches zero, a PCI interrupt can be generated. Additionally, the Interrupt Count register will be reloaded from the Interrupt Base Count and continue counting down for the next interrupt.

PCI Interrupts

There are a variety of potential sources of interrupts to the PCI host besides the bus master DMA interrupts. A single interrupt pin, $\overline{\text{INTA}}$ is used to signal these interrupts back to the host. The PCI Interrupt Register consolidates all of the possible interrupt sources; the bits of this register are shown in Table 5 on page 9. The register bits are set by the various sources, and can be cleared by writing a 1 to the bit(s) to be cleared.

PCI Control Register.

This register must be initialized by the DSP ROM code prior to PCI enumeration. (It has no effect in ISA or USB mode.) Once the Configuration Ready bit has been set to 1, the PCI Control Register becomes read-only, and further access by the DSP to configuration space is disallowed. The bigs of this register are shown in Table 6 on page 10.

Table 5. PCI Interrupt Register

Bit	Name	Comments
0	Reserved	Reserved
1	Rx0 DMA Channel Interrupt	Receive Channel 0 Bus Master Transactions
2	Rx1 DMA Channel Interrupt	Receive Channel 1 Bus Master Transactions

Table 5. PCI Interrupt Register (Continued)

Bit	Name	Comments
3	Tx0 DMA Channel Interrupt	Transmit Channel 0 Bus Master Transactions
4	Tx1 DMA Channel Interrupt	Transmit Channel 1 Bus Master Transactions
5	Incoming Mailbox 0 PCI Interrupt	PCI to DSP Mailbox 0 Transfer
6	Incoming Mailbox 1 PCI Interrupt	PCI to DSP Mailbox 1 Transfer
7	Outgoing Mailbox 0 PCI Interrupt	DSP to PCI Mailbox 0 Transfer
8	Outgoing Mailbox 1 PCI Interrupt	DSP to PCI Mailbox 1 Transfer
9	Reserved	
10	Reserved	
11	GPIO Wakeup	I/O Pin Initiated
12	AC'97 Wakeup	AC'97 Interface Initiated
13	PCI Master Abort Interrupt	PCI Interface Master Abort Detected
14	PCI Target Abort Interrupt	PCI Interface Target Abort Detected
15	Reserved	

Table 6. PCI Control Register

Bit	Name	Comments
1-0	PCI Functions Configured	00 = one PCI Function enabled, 01= two functions, 10= three functions
2	Configuration Ready	When 0, disables PCI accesses to the ADSP-2192 (terminated with Retry). Must be set to 1 by DSP ROM code after initializing configuration space. Once 1, cannot be written to 0.
15-3	Reserved	

ADSP-2192 PCI Configuration Space

The ADSP-2192 PCI Interface provides three separate configuration spaces, one for each possible function. This document describes the registers in each function, their reset condition, and how the three functions interact to access and control the ADSP-2192 hardware.

Similarities Between the Three PCI Functions

Each function contains a complete set of registers in the predefined header region as defined in the PCI Local Bus Specification Revision 2.2. In addition, each function contains the optional registers to support PCI Bus Power Management. Generally, registers that are unimplemented or read-only in one function are similarly defined in the other functions. Each function contains four base address registers that are used to access ADSP-2192 control registers and DSP memory.

Base address register (BAR) 1 is used to access the ADSP-2192 control registers. Accesses to the control registers via BAR1 uses PCI memory accesses. BAR1 requests a memory allocation of 1024 bytes. Access to DSP memory occurs via BAR2 and BAR3. BAR2 is used to access 24-bit DSP memory (for DSP program downloading) while BAR3 is used to access 16-bit DSP memory. BAR4 provides I/O space access to both the control registers and the DSP memory.

Table 7 on page 11 shows the configuration space headers for the three spaces. While these are the default uses for each of the configurations, they can be redefined to support any possible function by writing to the class code register of that function during boot. Additionally, during boot time, the DSP can disable one or more of the functions. If only two functions are enabled, they will be functions 0 and 1. If only one function is enabled, it will be function 0.

Interactions Between the Three PCI Configurations

Because the configurations must access and control a single set of resources, potential conflicts can occur between the control specified by the configuration.

Target accesses to registers and DSP memory can go through any function. As long as the Memory Space access enable bit is set in that function, then PCI memory accesses whose addresses match the locations programmed into a function, BARs 1-3 will be able to read or write any visible register or memory location within the ADSP-2192. Similarly, if IO Space access enable is set, then PCI I/O accesses can be performed via BAR4.

Within the Power Management section of the configuration blocks, there are a few interactions. The part will stay in the highest power state between the three configurations.

Table 7. PCI Configuration Space 0, 1, and 2

Address	Name	Reset	Comments
0x01-0x00	Vendor ID	0x11D4	Writable from the DSP during initialization
0x03-0x02	Config 0 Device ID	0x2192	Writable from the DSP during initialization
	Config 1 Device ID	0x219A	Writable from the DSP during initialization
	Config 2 Device ID	0x219E	Writable from the DSP during initialization
0x05-0x04	Command Register	0x0	Bus Master, Memory Space Capable, I/O Space Capable
0x07-0x06	Status Register	0x0	Bits enabled: Capabilities List, Fast B2B, Medium Decode
0x08	Revision ID	0x0	Writable from the DSP during initialization
0x0B-0x09	Class Code	0x48000	Writable from the DSP during initialization
0x0C	Cache Line Size	0x0	Read-only
0x0D	Latency Timer	0x0	
0x0E	Header Type	0x80	Multifunction bit set
0x0F	BIST	0x0	Unimplemented
0x13-0x10	Base Address1	0x08	Register Access for all ADSP-2192 Registers, Prefetchable Memory
0x17-0x14	Base Address2	0x08	24-bit DSP Memory Access
0x1B-0x18	Base Address3	0x08	16-bit DSP Memory Access
0x1F-0x1C	Base Address4	0x01	I/O access for control registers and DSP memory
0x23-0x20	Base Address5	0x0	Unimplemented
0x27-0x24	Base Address6	0x0	Unimplemented
0x2B- 0x28	Config 0 Cardbus CIS Pointer	0x1FF03	CIS RAM Pointer - Function 0 (Read Only).
	Config 1 Cardbus CIS Pointer	0x1FE03	CIS RAM Pointer - Function 1 (Read Only).
	Config 2 Cardbus CIS Pointer	0x1FD03	CIS RAM Pointer - Function 2 (Read Only).

Table 7. PCI Configuration Space 0, 1, and 2 (Continued)

Address	Name	Reset	Comments
0x2D- 0x2C	Subsystem Vendor ID	0x11D4	Writable from the DSP during initialization
0x2F- 0x2E	Config 0 Subsystem Device ID	0x2192	Writable from the DSP during initialization
	Config 1 Subsystem Device ID	0x219A	Writable from the DSP during initialization
	Config 2 Subsystem Device ID	0x219E	Writable from the DSP during initialization
0x33- 0x30	Expansion ROM Base Address	0x0	Unimplemented
0x34	Capabilities Pointer	0x40	Read-only
0x3C	Interrupt Line	0x0	
0x3D	Interrupt Pin	0x1	Uses $\overline{\text{INTA}}$ Pin
0x3E	Min_Gnt	0x1	Read-only
0x3F	Max_Lat	0x4	Read-only
0x40	Capability ID	0x1	Power Management Capability Identifier
0x41	Next_Cap_Ptr	0x0	Read-only
0x43- 0x42	Power Management Capabilities	0x6C22	Writable from the DSP during initialization
0x45- 0x44	Power Management Control/Status	0x0	Bits 15 and 8 initialized only on Power-up
0x46	Power Management Bridge	0x0	Unimplemented
0x47	Power Management Data	0x0	Unimplemented

ADSP-2192 PCI Memory Map

The ADSP-2192 On-Chip Memory is mapped to the PCI Address Space. Because some ADSP-2192 Memory Blocks are 24-bits wide (Program Memory) while others are 16-bits (Data Memory), two different footprints are available in PCI Address Space. These footprints are available to each PCI function by accessing different PCI Base Address Registers (BAR).

BAR2 supports 24-bit "Unpacked" Memory Access. BAR3 supports 16-bit "Packed" Memory Access.

In 24-bit (BAR2) Mode, each 32 bits (4 Consecutive PCI Byte Address Locations, which make up one PCI Data word) correspond to a single ADSP-2192 Memory Location. BAR2 Mode is typically used for Program Memory Access. Byte3 is always unused. Bytes[2:0] are used for 24-bit Memory Locations. Bytes[2:1] are used for 16-bit Memory Locations as shown in the example figure.

In 16-bit (BAR3) Mode, each 32-bit (4 Consecutive PCI Byte Address Locations) PCI Data word corresponds to two ADSP-2192 Memory Locations. Bytes[3:2] contain one 16-bit Data Word, Bytes[1:0] contain a second 16-bit Data Word. BAR3 Mode is typically used for Data Memory Access. Only the 16 MSBs of a Data Word are accessed in 24-bit Blocks; the 8 LSBs are ignored.

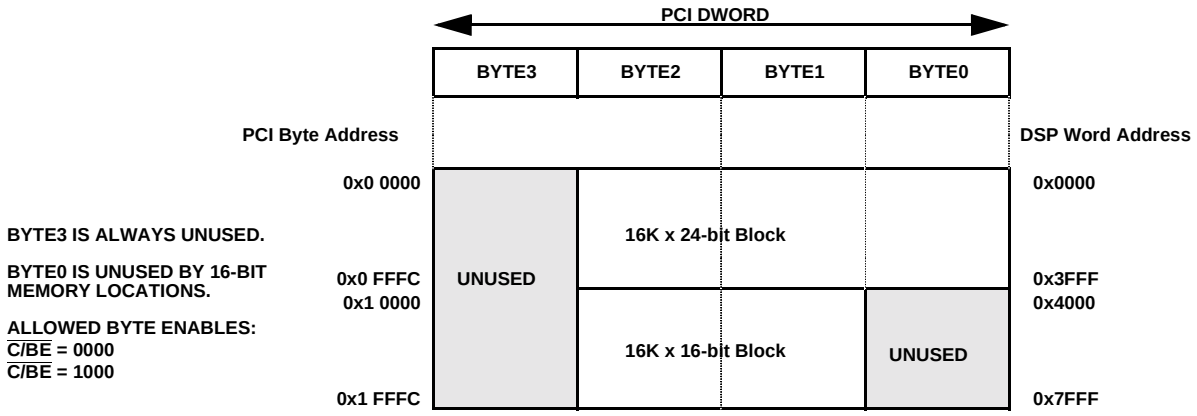

Figure 4. PCI Addressing for 24-bit and 16-bit Memory Blocks in 24-bit Access (BAR2) Mode

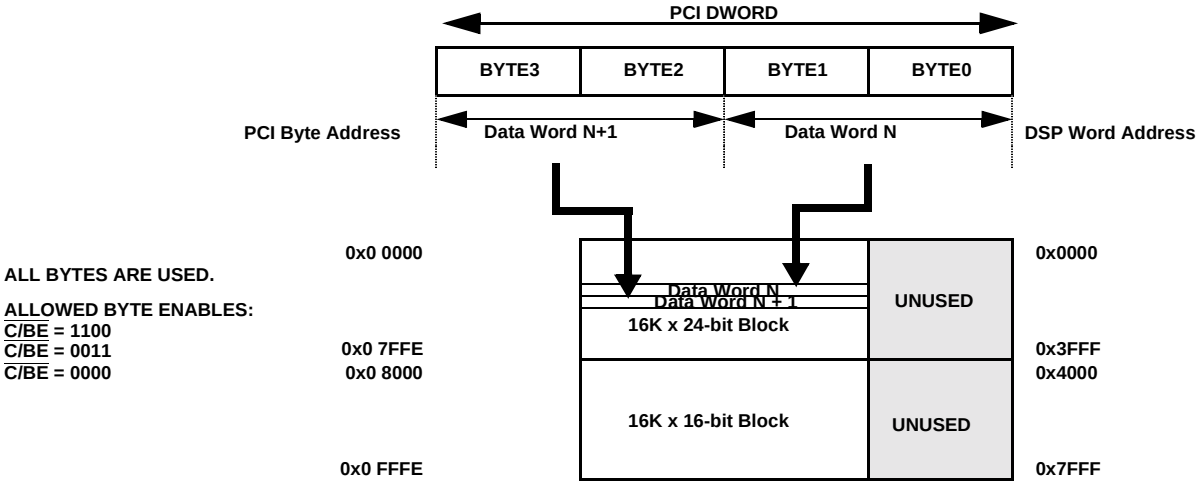

Figure 5. PCI Addressing for 24-bit and 16-bit Memory Blocks in 16-bit Access (BAR3) Mode.

24-bit PCI DSP Memory Map (BAR2)

The Complete PCI Address Footprint for the ADSP-2192 DSP Memory Spaces in 24-bit (BAR 2) Mode is as follows:

Table 8. 24-bit PCI DSP Memory Map (BAR 2 Mode)

Block	Byte3	Byte2	Byte1	Byte0	Offset
DSP P0 Data RAM Block 0	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0000 0000
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0000 0004
	:				
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0000 FFFC
DSP P0 Data RAM Block 1	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0001 0000
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0001 0004
	:				
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0001 FFFC
DSP P0 Data RAM Block 2	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0002 0000
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0002 0004
	:				
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0002 FFFC
DSP P0 Data RAM Block 3	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0003 0000
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0003 0004
	:				
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0003 FFFC
DSP P0 Program RAM Block	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x0004 0000
	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x0004 0004
	:				
	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x0004 FFFC
DSP P0 Program ROM Block	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x0005 0000
	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x0005 0004
	:				
	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x0005 3FFC
Reserved Space	RESERVED	RESERVED	RESERVED	RESERVED	0x0005 4000
	:				
	RESERVED	RESERVED	RESERVED	RESERVED	0x0007 FFFC
DSP P1 Data RAM Block 0	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0008 0000
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0008 0004
	:				
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0008 FFFC

Table 8. 24-bit PCI DSP Memory Map (BAR 2 Mode) (Continued)

Block	Byte3	Byte2	Byte1	Byte0	Offset
DSP P1 Data RAM Block 1	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0009 0000
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0009 0004
	:				
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x0009 FFFC
Reserved Space.	UNUSED	D[15:8]	D[7:0]	UNUSED	0x000A 0000
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x000A 0004
	:				
	UNUSED	D[15:8]	D[7:0]	UNUSED	0x000B FFFC
DSP P1 Program RAM Block	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x000C 0000
	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x000C 0004
	:				
	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x000C FFFC
DSP P1 Program ROM Block	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x000D 0000
	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x000D 0004
	:				
	UNUSED	D[23:16]	D[15:8]	D[7:0]	0x000D 3FFC
Reserved Space	RESERVED	RESERVED	RESERVED	RESERVED	0x000D 4000
	:				
	RESERVED	RESERVED	RESERVED	RESERVED	0x000F FFFC

16-bit PCI DSP Memory Map (BAR3)

The Complete PCI Address Footprint for the ADSP-2192 DSP Memory Spaces in 16-bit (BAR 3) Mode is as follows:

Table 9. 16-bit PCI DSP Memory Map (BAR 3 Mode)

Block	Byte3	Byte2	Byte1	Byte0	Offset
DSP P0 Data RAM Block 0	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0000 0000
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0000 0004
	:				
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0000 7FFC
DSP P0 Data RAM Block 1	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0000 8000
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0000 8004
	:				
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0000 FFFC
DSP P0 Data RAM Block 2	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0001 0000
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0001 0004
	:				
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0001 7FFC

Table 9. 16-bit PCI DSP Memory Map (BAR 3 Mode) (Continued)

Block	Byte3	Byte2	Byte1	Byte0	Offset
DSP P0 Data RAM Block 3	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0001 8000
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0001 8004
	:				
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0001 FFFC
DSP P0 Program RAM Block	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0002 0000
	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0002 0004
	:				
	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0002 7FFC
DSP P0 Program ROM Block	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0002 8000
	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0002 8004
	:				
	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0002 9FFC
Reserved Space	RESERVED	RESERVED	RESERVED	RESERVED	0x0002 A000
	:				
	RESERVED	RESERVED	RESERVED	RESERVED	0x0003 FFFC
DSP P1 Data RAM Block 0	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0004 0000
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0004 0004
	:				
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0004 7FFC
DSP P1 Data RAM Block 1	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0004 8000
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0004 8004
	:				
	D[15:8]	D[7:0]	D[15:8]	D[7:0]	0x0004 FFFC
Reserved Space.	RESERVED	RESERVED	RESERVED	RESERVED	0x0005 0000
	:				
	RESERVED	RESERVED	RESERVED	RESERVED	0x0005 FFFC
DSP P1 Program RAM Block	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0006 0000
	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0006 0004
	:				
	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0006 7FFC
DSP P1 Program ROM Block	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0006 8000
	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0006 8004
	:				
	D[23:16]	D[15:8]	D[23:16]	D[15:8]	0x0006 9FFC
Reserved Space	RESERVED	RESERVED	RESERVED	RESERVED	0x0006 A000
	:				
	RESERVED	RESERVED	RESERVED	RESERVED	0x0007 FFFC

16-bit PCI DSP I/O Memory Map (BAR4)

PCI Base Address Register (BAR 4) allows indirect access to the ADSP-2192 Control Registers and DSP Memory. The DSP Memory Indirect Access Registers accessible from BAR4 are as follows:

Table 10. 16-bit PCI DSP I/O Space Indirect Access Registers Map (BAR 4 Mode)

Offset	Name	Reset	Comments
0x03-0x00	Control Register Address	0x0000	Address and direction control for registers accesses
0x07-0x04	Control Register Data	0x0000	Data for register accesses
0x0B-0x08	DSP Memory Address	0x000000	Address and Direction control for Indirect DSP memory accesses
0x0F-0x0C	DSP Memory Data	0x000000	Data for DSP memory accesses

DSP P0 Memory Indirect Address Space occupies PCI BAR4 Space 0x000000 -> 0x01FFFF

DSP P1 Memory Indirect Address Space occupies PCI BAR4 Space 0x020000 -> 0x03FFFF

All Indirect DSP Memory Accesses are 24-bit or 16-bit Word Accesses.

Using the USB Interface

The ADSP-2192 USB design enables the ADSP-2192 to be configured and attached to a single device with multiple interfaces and various endpoint configurations, as follows:

1. Programmable descriptors and a class-specific command interpreter are accessible through the USB 8052 registers. An 8052-compatible MCU is supported on-board, to enable soft downloading of different configurations, and support of standard or class-specific commands.
2. A total of 8 user-defined endpoints are provided. Endpoints can be configured as BULK, ISO, or INT, and can be grouped

USB DSP Register Definitions

For each endpoint, four registers are defined to provide a memory buffer in the DSP. These registers are defined for each endpoint shared by all defined interfaces, for a total of 4x8 = 32 registers. These registers are read/write by the DSP only.

Table 11. USB DSP Register Definitions

Page	Address	Name	Comment
0x0C	0x0-0x3	DSP Memory Buffer Base Addr	EP4
0x0C	0x4-0x5	DSP Memory Buffer Size	EP4
0x0C	0x6-0x7	DSP Memory Buffer RD Offset	EP4
0x0C	0x8-0x9	DSP Memory Buffer WR Offset	EP4
0x0C	0x10-0x13	DSP Memory Buffer Base Addr	EP5
0x0C	0x14-0x15	DSP Memory Buffer Size	EP5
0x0C	0x16-0x17	DSP Memory Buffer RD Offset	EP5

Table 11. USB DSP Register Definitions (Continued)

Page	Address	Name	Comment
0x0C	0x18-0x19	DSP Memory Buffer WR Offset	EP5
0x0C	0x20-0x23	DSP Memory Buffer Base Addr	EP6
0x0C	0x24-0x25	DSP Memory Buffer Size	EP6
0x0C	0x26-0x27	DSP Memory Buffer RD Offset	EP6
0x0C	0x28-0x29	DSP Memory Buffer WR Offset	EP6
0x0C	0x30-0x33	DSP Memory Buffer Base Addr	EP7
0x0C	0x34-0x35	DSP Memory Buffer Size	EP7
0x0C	0x36-0x37	DSP Memory Buffer RD Offset	EP7
0x0C	0x38-0x39	DSP Memory Buffer WR Offset	EP7
0x0C	0x40-0x43	DSP Memory Buffer Base Addr	EP8
0x0C	0x44-0x45	DSP Memory Buffer Size	EP8
0x0C	0x46-0x47	DSP Memory Buffer RD Offset	EP8
0x0C	0x48-0x49	DSP Memory Buffer WR Offset	EP8
0x0C	0x50-0x53	DSP Memory Buffer Base Addr	EP9
0x0C	0x54-0x55	DSP Memory Buffer Size	EP9
0x0C	0x56-0x57	DSP Memory Buffer RD Offset	EP9
0x0C	0x58-0x59	DSP Memory Buffer WR Offset	EP9
0x0C	0x60-0x63	DSP Memory Buffer Base Addr	EP10
0x0C	0x64-0x65	DSP Memory Buffer Size	EP10
0x0C	0x66-0x67	DSP Memory Buffer RD Offset	EP10
0x0C	0x68-0x69	DSP Memory Buffer WR Offset	EP10
0x0C	0x70-0x73	DSP Memory Buffer Base Addr	EP11
0x0C	0x74-0x75	DSP Memory Buffer Size	EP11
0x0C	0x76-0x77	DSP Memory Buffer RD Offset	EP11
0x0C	0x78-0x79	DSP Memory Buffer WR Offset	EP11
0x0C	0x80-0x81	USB Descriptor Vendor ID	
0x0C	0x84-0x85	USB Descriptor Product ID	
0x0C	0x86-0x87	USB Descriptor Release Number	
0x0C	0x88-0x89	USB Descriptor Device Attributes	

USB DSP Memory Buffer Base Addr Register

Points to the base address for the DSP memory buffer assigned to this endpoint.

- BA[17:0] = Memory Buffer Base Address

USB DSP Memory Buffer Size Register

Indicates the size of the DSP memory buffer assigned to this endpoint.

- SZ[15:0] = Memory Buffer Size

USB DSP Memory Buffer RD Pointer Offset Register

The offset from the base address for the read pointer of the memory buffer assigned to this endpoint.

- RD[15:0] = Memory Buffer RD Offset

USB DSP Memory Buffer WR Pointer Offset Register

The offset from the base address for the write pointer of the memory buffer assigned to this endpoint.

- WR[15:0] = Memory Buffer WR Offset

USB Descriptor Vendor ID

The Vendor ID returned in the GET DEVICE DESCRIPTOR command is contained in this register. The DSP can change the Vendor ID by writing to this register during the Serial EEPROM initialization. The default Vendor ID is 0x0456, which corresponds to Analog Devices, Inc.

- V[15:0] = Vendor ID (default = 0x0456)

USB Descriptor Product ID

The Product ID returned in the GET DEVICE DESCRIPTOR command is contained in this register. The DSP can change the Product ID by writing to this register during the Serial EEPROM initialization. The default Product ID is 0x2192.

- P[15:0] = Product ID (default = 0x2192)

USB Descriptor Release Number

The Release Number returned in the GET DEVICE DESCRIPTOR command is contained in this register. The DSP can change the Release Number by writing to this register during the Serial EEPROM initialization. The default Release Number is 0x0100, which corresponds to Version 01.00.

- R[15:0] = Release Number (default = 0x0100)

USB Descriptor Device Attributes

The device-specific attributes returned in the GET DEVICE DESCRIPTOR command are contained in this register. The DSP can change the attributes by writing to this register during the Serial EEPROM initialization. The default attributes are 0x80FA, which correspond to bus-powered, no remote wake-up, and max power = 500mA.

- SP: 1=self-powered, 0=bus-powered (default = 0)
- RW: 1=have remote wake-up capability, 0=no remote wake-up capability (default = 0)
- C[7:0] = power consumption from bus, expressed in 2mA units (default = 0xFA 500mA)

USB DSP MCU Register Definitions

MCU registers are defined in four memory spaces that are grouped by the following address ranges:

- 0x0XXX—This address range defines general purpose USB status and control registers
- 0x1XXX—This address range defines registers that are specific to endpoint setup and control
- 0x2XXX—This address range defines the registers used for REGIO accesses to the DSP register space
- 0x3XXX—This address range defines the MCU program memory write address space

Table 12. USB MCU Register Definitions

Address	Name	Comments
0x0000- 0x0007	USB SETUP Token Cmd	8 bytes total
0x0008- 0x000F	USB SETUP Token Data	8 bytes total
0x0010- 0x0011	USB SETUP Counter	16 bit counter
0x0012- 0x0013	USB Control	Miscellaneous control including re-attach
0x0014- 0x0015	USB Address/Endpoint	Address of device/active endpoint
0x0016- 0x0017	USB Frame Number	Current frame number
0x0030- 0x0031	USB Serial EEPROM Mailbox 1	Defined by ADI
0x0032- 0x0033	USB Serial EEPROM Mailbox 2	Defined by ADI
0x0034- 0x0035	USB Serial EEPROM Mailbox 3	Defined by ADI

Table 12. USB MCU Register Definitions (Continued)

Address	Name	Comments
0x1000- 0x1001	USB EP4 Description	Configures endpoint
0x1002- 0x1003	USB EP4 NAK	Counter
0x1004- 0x1005	USB EP5 Description	Configures endpoint
0x1006- 0x1007	USB EP5 NAK	Counter
0x1008- 0x1009	USB EP6 Description	Configures endpoint
0x100A- 0x100B	USB EP6 NAK	Counter
0x100C- 0x100D	USB EP7 Description	Configures endpoint
0x100E- 0x100F	USB EP7 NAK	Counter
0x1010- 0x1011	USB EP8 Description	Configures endpoint
0x1012- 0x1013	USB EP8 NAK	Counter
0x1014- 0x1015	USB EP8 Description	Configures endpoint
0x1016- 0x1017	USB EP9 NAK	Counter
0x1018- 0x1019	USB EP10 Description	Configures endpoint
0x101A- 0x101B	USB EP10 NAK	Counter
0x101C- 0x101D	USB EP11 Description	Configures endpoint
0x101E- 0x101F	USB EP11 NAK	Counter
0x1020- 0x1021	USB EP STALL	Policy
0x1040- 0x1043	USB EP1 Code Download Base Address	Starting address for code download on endpoint 1
0x1044- 0x1047	USB EP2 Code Download Base Address	Starting address for code download on endpoint 2
0x1048- 0x104B	USB EP3 Code Download Base Address	Starting address for code download on endpoint 3
0x1060- 0x1063	USB EP1 Code Current Write Pointer Offset	Current write pointer offset for code download on endpoint 1
0x1064- 0x1067	USB EP2 Code Current Write Pointer Offset	Current write pointer offset for code download on endpoint 2
0x1068- 0x106B	USB EP3 Code Current Write Pointer Offset	Current write pointer offset for code download on endpoint 3
0x2000- 0x2001	USB Register I/O Address	
0x2002- 0x2003	USB Register I/O Data	
0x3000- 0x3FFF	USB MCU Program Memory	

USB Endpoint Description Register

The endpoint description register provides the USB core with information about the endpoint type, direction, and max packet size. This register is read/write by the MCU only. This register is defined for endpoints 4-11.

- PS[9:0] MAX Packet Size for endpoint
- LT[1:0] Last transaction indicator bits: 00 = Clear, 01 = ACK, 10 = NAK, or 11 = ERR
- TY[1:0] Endpoint type bits: 00 = DISABLED, 01 = ISO, 10 = Bulk, or 11 = Interrupt
- DR Endpoint direction bit: 1 = IN or 0 = OUT
- TB Toggle bit for endpoint. Reflects the current state of the DATA toggle bit.

USB Endpoint NAK Counter Register

This register records the number of sequential NAKs that have occurred on a given endpoint. This register is defined for endpoints 4-11. This register is read/write by the MCU only.

- N[3:0] NAK counter. Number of sequential NAKs that have occurred on a given endpoint. When N[3:0] is equal to the base NAK counter NK[3:0], a zero-length packet or packet less that maxpacketsize will be issued.
- ST 1 = Endpoint is stalled

USB Endpoint Stall Policy Register

This register contains NAK count and endpoint FIFO error policy bit. The STALL status bits for endpoints 1-3 are included as well. This register is read/write by the MCU only.

- ST[3:1] 1 = Endpoint is stalled. ST[1] maps to endpoint 1, ST[2] maps to endpoint 2, etc.
- NK[3:0] Base NAK counter. Determines how many sequential NAKs are issued before sending zero length packet on any given endpoint.
- FE FIFO error policy. 1 = When endpoint FIFO is overrun/underrun, STALL endpoint

USB Endpoint 1 Code Download Base Address Register

This register contains an 18 bit address which corresponds to the starting location for DSP code download on endpoint 1. This register is read/write by the MCU only.

USB Endpoint 2 Code Download Base Address Register

This register contains an 18 bit address which corresponds to the starting location for DSP code download on endpoint 2. This register is read/write by the MCU only.

USB Endpoint 3 Code Download Base Address Register

This register contains an 18 bit address which corresponds to the starting location for DSP code download on endpoint 3. This register is read/write by the MCU only.

USB Endpoint 1 Code Current Write Pointer Offset Register

This register contains an 18 bit address which corresponds to the current write pointer offset from the base address register for DSP code download on endpoint 1. The sum of this register and the EP1 Code Download Base Address Register represents the last DSP PM location written.

This register is read by the MCU only and is cleared to 3FFFF (-1) when the Endpoint 1 Code Download Base Address Register is updated.

USB Endpoint 2 Code Current Write Pointer Offset Register

This register contains an 18 bit address which corresponds to the current write pointer offset from the base address register for DSP code download on endpoint 2. The sum of this register and the EP2 Code Download Base Address Register represents the last DSP PM location written.

This register is read by the MCU only and is cleared to 3FFFF (-1) when the Endpoint 2Code Download Base Address Register is updated.

USB Endpoint 3 Code Current Write Pointer Offset Register

This register contains an 18 bit address which corresponds to the current write pointer offset from the base address register for DSP code download on endpoint 3. The sum of this register and the EP3 Code Download Base Address Register represents the last DSP PM location written.

This register is read by the MCU only and is cleared to 3FFFF (-1) when the Endpoint 3Code Download Base Address Register is updated.

USB SETUP Token Command Register

This register is defined as 8 bytes long and contains the data sent on the USB from the most recent SETUP transaction. This register is read by the MCU only.

USB SETUP Token Data Register

If the most recent SETUP transaction involves a data OUT stage, this register is defined as 8 bytes long and contains the data sent on the USB during the data stage. This is also where the MCU will write data to be sent in response to a SETUP transaction involving a data IN stage. This register is read/write by the MCU only.

USB SETUP Counter Register

This register provides information as the total size of the setup transaction data stage. This register is read/write by the MCU only.

- C[3:0] Total amount of data (bytes) to be sent/received during the data stage of the SETUP transaction

USB Register I/O Address Register

This register contains the address of the ADSP-2192 register that is to be read/written. This register is read/write by the MCU only.

- A[15] Start ADSP-2192 read/write cycle
- A[14] 1 = WRITE, 0 = READ
- A[13:0] ADSP-2192 address to read/write

USB Register I/O Data Register

This register contains the data of the ADSP-2192 register which has been read or is to be written. This register is read/write by the MCU only.

- D[15:0] During READ this register contains the data read from the ADSP-2192, during WRITE this register is the data to be written to the ADSP-2192

USB Control Register

This register controls various USB functions. This register is read/write by the MCU only.

- MO 1 = MCU has completed boot sequence and is ready to respond to USB commands
- DI 1 = Disconnect CONFIG device and enumerate again using the downloaded MCU configuration
- BB 1 = After reset boot from MCU RAM; 0 = after reset boot from MCU ROM
- INT = Active interrupt for the 8052 MCU
- ISE = Current interrupt is for a SETUP token
- IIN = Current interrupt is for an IN token
- IOU = Current interrupt is for an OUT token
- ER = Error in the current SETUP transaction. Generate STALL condition on EP0.

USB Address/Endpoint Register

This register contains the USB address and active endpoint. This register is read/write by the MCU only.

- A[6:0] USB address assigned to device
- EP[3:0] USB last active endpoint

USB Frame Number Register

This register contains the last USB frame number. This register is read by the MCU only.

- FN[10:0] USB frame number

General USB Device Definitions

These definitions define the USB device descriptor, device config, and device endpoints.

CONFIG DEVICE DEFINITION

- FIXED ENDPOINTS
- CONTROL ENDPOINT 0
- Type: Control
- Dir: Bi-directional
- Maxpacketsize: 8

CONFIG DEVICE Device Descriptor

Note: Offset fields 8-13 are user-definable via Serial EEPROM

Table 13. CONFIG DEVICE Device Descriptor

Offset	Field	Description	Value
0	bLength	Length = 18 bytes	0x12
1	bDescriptorType	Type = DEVICE	0x01
2 - 3	bcdUSB	USB Spec 1.1	0x0110
4	bDeviceClass	Device class vendor specific	0xFF
5	bDeviceSubClass	Device sub-class vendor specific	0xFF
6	bDeviceProtocol	Device protocol vendor specific	0xFF
7	bMaxPacketSize	Max packet size for EP0 = 8 bytes	0x08
8 - 9	idVendor (L)	Vendor ID (L) = 0456 ADI	0x0456
10 - 11	idProduct (L)	Product ID (L) = ADSP-2192	0x2192
12 - 13	bcdDevice (L)	Device release number = 1.00	0x0100

Table 13. CONFIG DEVICE Device Descriptor (Continued)

Offset	Field	Description	Value
14	iManufacturer	Manufacturer index string	0x01
15	iProduct	Product index string	0x02
16	iSerialNumber	Serial number index string	0x00
17	bNumConfigurations	Number of configurations = 1	0x01

Table 14. CONFIG DEVICE Configuration Descriptor

Offset	Field	Description	Value
0	bLength	Descriptor Length = 9 bytes	0x09
1	bDescriptorType	Descriptor Type = Configuration	0x02
2	wTotalLength (L)	Total Length (L)	0x12
3	wTotalLength (H)	Total Length (H)	0x00
4	bNumInterfaces	Number of Interfaces	0x01
5	bConfigurationValue	Configuration Value	0x01
6	iConfiguration	Index of string descriptor (None)	0x00
7	bmAttributes	Bus powered, no wake-up	0x80
8	MaxPower	Max power = 500mA	0xFA

Note: Offset fields 7-8 are user definable via Serial EEPROM

Table 15. CONFIG DEVICE String Descriptor Index 0

Offset	Field	Description	Value
0	bLength	Descriptor Length = 4 bytes	0x04
1	bDescriptorType	Descriptor Type = String	0x03
2	wLANGID[0]	LangID = 0409 (US English)	0x0409

Table 16. CONFIG DEVICE Descriptor Index 1 (Manufacturer)

Offset	Field	Description	Value
0	bLength	Descriptor Length = 20 bytes	0x14
1	bDescriptorType	Descriptor Type = String	0x03
2-19	bString	ADI	

Table 17. CONFIG DEVICE String Descriptor Index 2 (Product)

Offset	Field	Description	Value
0	bLength	Descriptor Length = 34 bytes	0x22
1	bDescriptorType	Descriptor Type = String	0x03
2-31	bString	ADI USB Device	

Configuration 0, 1, and 2 Device Definition

- FIXED ENDPOINTS
- CONTROL ENDPOINT 0
- Type: Control
- Dir: Bi-directional
- Maxpacketsize: 8
- BULK OUT ENDPOINT 1, 2, 3 = Used for code download to DSP
- Type: Bulk
- Dir: OUT
- Maxpacketsize: 64
- PROGRAMMABLE ENDPOINTS: 4 5 6 7 8 9 10 11
- Programmable in:
- Type: via USB Endpoint Description Register
- Direction: via USB Endpoint Description Register
- Maxpacket size: via USB Endpoint Description Register
- Memory Allocation: via DSP Memory Buffer Base Addr, DSP Memory Buffer Size, DSP Memory Buffer RD Pointer Offset, DSP Memory Buffer Write Pointer Offset Registers

Note: The GENERIC endpoints are shared between all interfaces.

Endpoint 0 Definition

In addition to the normally defined USB standard device requests, the following vendor specific device requests are supported with the use of EP0. These requests are issued from the host driver via normal SETUP transactions on the USB.

USB MCU Code Download

Address <15:0> is the first address to begin code download to; the address is incremented automatically after each byte is written. USB MCUCODE is a three-stage control transfer with an OUT data stage. Stage 1 is the SETUP stage, stage 2 is the data stage involving the OUT packet, and stage 3 is the status stage. The length of the data stage is determined by the driver and is specified by the total length of the MCU code to be downloaded. See Table 18 on page 24 for details about the USB MCUCODE (code download) fields.

USB REGIO (Write)

Address <15:15> = 1 indicates a write to the MCU register space; Address <15:15> = 0 indicates a write to the DSP register space. When accessing DSP register space, the MCU must write the data to be written into the USB Register I/O Data register and write the address to be written to the USB Register I/O Address register. Bit 15 of the USB Register I/O Address register starts the transaction and bit 14 is set to one to indicate a WRITE.

USB REGIO (register write) is a three-stage control transfer with an OUT data stage. Stage 1 is the SETUP stage, stage 2 is the data stage involving the OUT packet, and stage 3 is the status stage. See Table 19 on page 25 for details about the USB REGIO (register write) fields.

Table 18. USB MCUCODE (Code Download)

Offset	Field	Size	Value	Description
0	bmRequest	1	0x40	Vendor Request, OUT
1	bRequest	1	0xA1	USB MCUCODE
2	wValue (L)	1	XXX	Address <0:7>
3	wValue (H)	1	XXX	Address <8:15>
4	wIndex (L)	1	0x00	

Table 18. USB MCUCODE (Code Download) (Continued)

Offset	Field	Size	Value	Description
5	wIndex (H)	1	0x00	
6	wLength (L)	1	0xXX[1]	Length = XX bytes
7	wLength (H)	1	0xYY[2]	Length = YY bytes

[1] XX is user-specified.

[2] YY is user-specified.

Table 19. USB REGIO (Register Write)

Offset	Field	Size	Value	Description
0	bmRequest	1	0x40	Vendor Request, OUT
1	bRequest	1	0xA0	USB REGIO
2	wValue (L)	1	XXX	Address <0:7>
3	wValue (H)	1	XXX	Address <8:15>
4	wIndex (L)	1	0x00	
5	wIndex (H)	1	0x00	
6	wLength (L)	1	0x02	Length = 02 bytes
7	wLength (H)	1	0x00	

USB REGIO (Read)

Address <15:15> = 1 indicates a read to the MCU register space; Address <15:15> = 0 indicates a read to the DSP register space. When accessing DSP register space, the MCU must write the address to be read to the USB Register I/O Address register.

Bit 15 of the USB Register I/O Address register starts the transaction, and bit 14 is set to zero to indicate a READ. The data read will be placed into the USB Register I/O Data register.

USB REGIO (register read) is a three-stage control transfer with an IN data stage. Stage 1 is the SETUP stage, stage 2 is the data stage involving the IN packet, and stage 3 is the status stage. See Table 20 on page 25 for details about the USB REGIO (register read) fields.

DSP Code Download

Since EP0 only has a max packet size of 8, downloading DSP code on EP0 can be inefficient when operating on a UHCI controller which only allows fixed amount of control transactions per frame. Therefore, to gain better throughput for code download, downloading of DSP code involves synchronizing a control SETUP command on EP0 with BULK OUT commands on endpoints 1, 2, or 3. Each endpoint has an associated DSP download address that is set by using USB REGIO (Write) command.

Table 20. USB REGIO (Register Read)

Offset	Field	Size	Value	Description
0	bmRequest	1	0xC0	Vendor Request, IN
1	bRequest	1	0xA0	USB REGIO
2	wValue (L)	1	XXX	Address <0:7>

Table 20. USB REGIO (Register Read) (Continued)

Offset	Field	Size	Value	Description
3	wValue (H)	1	XXX	Address <8:15>
4	wIndex (L)	1	0x00	
5	wIndex (H)	1	0x00	
6	wLength (L)	1	0x02	Length = 02 bytes
7	wLength (H)	1	0x00	

Since there are three possible interfaces supported, each interface has its own DSP download address and uses its own BULK pipe to download code. The driver for each interface must set the download address before beginning to use the BULK pipe to download DSP code. The download address will auto-increment as each byte of data is sent on the BULK pipe to the DSP.

DSP instructions are three bytes long, and USB BULK pipes have even-number packet sizes. The instructions to be downloaded must be formatted into four-byte groups with the least significant byte always zero. The USB interface strips off the least significant byte and formats the DSP instruction properly before writing it into the program memory. For example, to write the three-byte opcode 0x400000 to DSP program memory, the driver sends 0x40000000 down the BULK pipe.

The following example illustrates the proper order of commands and synchronizing that the driver must follow.

1. Device enumerates with two interfaces. Each interface has the capability to download DSP code and can initiate at any time.
2. The driver for interface 1 begins code download by sending the USB REGIO (Write) command with the starting download address.

   The driver must wait for this command to finish before starting code download.
3. The driver for interface 2 begins code download by sending the USB REGIO (Write) command with the starting download address.

   The driver must wait for this command to finish before starting code download.
4. Each driver now streams the code to be downloaded to the DSP: driver 1 onto BULK EP1 for interface 1, and driver 2 onto BULK EP2 for interface 2. The code is written to the DSP in 3-byte instructions starting at the location specified by the USB REGIO (Write) command. The driver must wait for each command to finish before sending a new code download address.
5. If there is more code to be downloaded at a different starting address, the driver begins the entire sequence again, using steps 1-4.

General Comments:

- DSP code download is only available after the ADSP-2192 has re-enumerated using the MCU soft firmware. The DSP code download command will not be available in the MCU boot ROM for the default CONFIG device.
- After setting the download addresses using the USB REGIO (Write) command, code download can be initiated for any length using normal BULK traffic.

Example Initialization Process

After attachment to the USB bus, the ADSP-2192 identifies itself as a CONFIG device with one endpoint(s), which refers to its one control, EP0. This will cause a generic user-defined CONFIG driver to load.

The CONFIG driver downloads appropriate MCU code to setup the MCU, which includes the specific device descriptors, interfaces, and endpoints.

The external Serial EEPROM is read by the DSP and transferred to the MCU. The CONFIG driver through the control EP0 pipe generates a register read to determine the configuration value. Based on this configuration code, the host downloads the proper USB configurations to the MCU.

Finally the driver writes the USB Control Register, causing the device to disconnect and then reconnect so the new downloaded configuration is enumerated by the system. Upon enumeration, each interface loads the appropriate device driver.

An example of this procedure is configuring the ADSP-2192 to be an ADSL modem and a FAX modem.

1. ADSP-2192 device is attached to USB bus. System enumerates the CONFIG device in the ADSP-2192 first. A user-defined driver is loaded.
2. The user-defined driver reads the device descriptor, which identifies the card as an ADSL/FAX modem.
3. The user-defined driver downloads USB configuration and MCU code to the MCU for interface 1, which is the ADSL modem.

4. Configuration specifies which endpoints are used (and their definitions). A typical configuration for ADSL appears in Table 21.
5. The user-defined driver downloads USB configuration for interface 2, which is the FAX modem. Configuration specifies which endpoints are used and their definitions. A typical configuration for FAX appears in Table 22.
6. The user-defined driver now writes the USB Config Register, which causes the device to disconnect and reconnect. The system enumerates all interfaces and loads the appropriate drivers.
7. ADSL driver downloads code to DSP for ADSL service. DSP also initializes the USB Endpoint Description Register, DSP Memory Buffer Base Addr Register, DSP Memory Buffer Size Register, DSP Memory Buffer RD Pointer Offset, and DSP Memory Buffer WR Pointer Offset registers for each endpoint. Endpoints can only be used when these registers have been written. ADSL service is now available.
8. FAX driver downloads code to DSP for FAX service. DSP also initializes the USB Endpoint Description Register, DSP Memory Buffer Base Addr Register, DSP Memory Buffer Size Register, DSP Memory Buffer RD Pointer Offset, and DSP Memory Buffer WR Pointer Offset registers for each endpoint. Endpoints can only be used when the above registers have been written. FAX service is now available.

ADSP-2192 USB Data Pipe Operations

All data transactions involving the generic endpoints (4-11) stream data into and out of the DSP memory via a dedicated USB hardware block. This hardware block manages all USB transactions for these endpoints and serves as a conduit for the data moving to and from the DSP memory FIFOs. There is no MCU involvement in the management of these data pipes.

Table 21. Typical Configuration for ADSL Modem

End Point	Type	Max Packet	Comment
1	BULK OUT	64	DSP CODE
4	BULK IN	64	ADSL RCV
5	BULK OUT	64	ADSL XMT
6	INT IN	16	STATUS

Table 22. Typical Configuration for FAX Modem

End Point	Type	Max Packet	Comment
2	BULK OUT	64	DSP CODE
7	BULK IN	64	FAX RCV
8	BULK OUT	64	FAX XMT
9	INT IN	16	STATUS

The USB data FIFOs for these generic endpoints exist in DSP memory space. For each endpoint, there exist the following memory buffer registers:

- Base Address (18 bits)
- Size (16 bits) - Offset from the Base Address
- Read Offset (16 bits) - Offset from the Base Address
- Write Offset (16 bits) - Offset from the Base Address

As part of initialization, the DSP code sets up these FIFOs before USB data transactions for these endpoints can begin. DSP memory addresses cannot exceed 18 bits (0x000000 - 0x03FFFF). When setting up these USB FIFOs, Base+Size/Read Off-set/ Write Offset cannot be greater than 18 bits.

The DSP memory interface on the ADSP-2192 only allows reads/writes of 16-bit words. It cannot handle byte transactions. Therefore, a 64 byte maxpacketsize means 32 DSP words. A single byte cannot be transferred to/from the DSP. Endpoint 0 does not have this limitation. Since these FIFOs exist in DSP memory, the DSP shares some pointer management tasks with the USB core. For OUT transactions, the write pointer is controlled by the USB core, while the read pointer is governed by the DSP. The opposite is true for IN transactions.

Both the write and read pointers for each memory buffer would start off at zero. All USB buffers operate in a circular fashion. Once a pointer reaches the end of the buffer, it will need to be set back to zero.

OUT Transactions (Host -> Device)

When an OUT transaction arrives for a particular endpoint, the USB core calculates the difference between the write and read pointers to determine the amount of room available in the FIFOs. If all of the OUT data arrives and the write pointer never catches up to the read pointer, that data is Backed and the USB core updates the Memory Buffer Write Offset register.

If at any time during the transaction the two pointers collide, the USB block responds with a NAK indicating that the host must re-send the same data packet; in that case, the write pointer remains unchanged.

If for some reason the host sends more data than the maxpacketsize, the USB core accepts it, as long as there is sufficient room in the FIFO.

Since the DSP controls the read pointer, it must perform a similar calculation to determine if there is sufficient data in the FIFO to begin processing. Once it has consumed some amount of data, the DSP will need to update the Memory Buffer Read Offset register.

IN Transactions (Host <- Device)

When an IN transaction arrives for a particular endpoint, the USB core once again computes how much read data is available in the FIFO. It also determines if the amount of read data is greater than or equal to the maxpacketsize. If both conditions are met, the USB core will transfer the data. Upon receiving ACK from the host, the USB core updates the Memory Buffer Read Offset register.

If the amount of read data is less than the maxpacketsize (a short packet), the USB core determines whether to send the data based upon a NAK count limit. This is a 4-bit field in the Endpoint Stall Policy register that can be programmed with a value indicating how many NAK's should be sent prior to transmitting a short packet. This allows flexibility in determining how IRPs are retired via short packets.

Since the DSP controls the write pointer, it must determine if there is sufficient room in the FIFO for placing new data. Once it has completed writes to the FIFO, it needs to update the Memory Buffer Write Offset register.

Sub-ISA Interface

In systems which combine the ADSP-2192 chip with other devices on a single PCI interface, the ADSP-2192 Sub-ISA mode is used to provide a simpler interface (to a PCI function ASIC), which bypasses the ADSP-2192's PCI interface.

In this mode, the Combo Master assumes all responsibility for interfacing the function to the PCI bus, including provision of Configuration Space registers for the ADSP-2192 system as a separate PnP function. In Sub-ISA Mode the PCI Pins are reconfigured for ISA operation, as follows.

Table 23. Sub-ISA (PCI) Pin Descriptions

Pin Name	PCI Direction[1]	ISA Alias	ISA Direction	ISA Description.
AD[15:0]	In/Out	ISAD[15:0]	In/Out	Data
AD[18:16]	In/Out	ISAA[3:1]	In	Register Address
AD[31:22]	In/Out	Unused	In	Tie to GND in Sub-ISA Mode
$\overline{\text{RST}}$	In	$\overline{\text{RST}}$	In	Reset
$\overline{\text{CBE0}}$	In/Out	$\overline{\text{IOW}}$	In	Write Strobe
$\overline{\text{CBE1}}$	In/Out	$\overline{\text{IOR}}$	In	Read Strobe
$\overline{\text{CBE2}}$	In/Out	$\overline{\text{AEN}}$	In	Chip Select (Access Enable)
$\overline{\text{INTA}}$	Out (o/d)	IRQ	Out	(CMOS) Interrupt (Active High)
AD21	In/Out	$\overline{\text{PDW1}}$	In	PCI D-state MSB (inverted) Power-Down
AD20	In/Out	$\overline{\text{PDW0}}$	In	PCI D-state LSB (inverted) Power-Down
AD19	In/Out	PME_EN	In	PME Enable
$\overline{\text{PME}}$	Out (o/d)	$\overline{\text{PMERQ}}$	Out (o/d)	Power Management Event
CLK	In	Unused	In	Tie to GND in Sub-ISA Mode
$\overline{\text{CLKRUN}}$	In/Out	IOCHRDY	Out	IO Ready
CLKRUN	Out	IOCHRDY	Out	Acknowledge

[1] o/d = Open Drain

In Sub-ISA mode, the ADSP-2192's PCI protocol is replaced with an ISA-like, asynchronous protocol controlled by the strobes $\overline{\text{IOR}}$, $\overline{\text{IOW}}$ and $\overline{\text{AEN}}$. Access is possible only to the PCI Base Address 4 (BAR4) Registers (the InDirect Access Registers). The Sub-ISA Address Map is shown in Table 23 on page 28.

An active low $\overline{RST}$ input (to be derived from PCI $\overline{RST}$ and possible other sources) and an active-high IRQ interrupt output are available. Power Management is handled by the ADSP-2192 inputs $\overline{PDW1–0}$/PME_EN and the ADSP-2192 output $\overline{PMERQ}$. $\overline{PDW1–0}$ should be the inversion of the PCI power state in the function's PMCSR register. $\overline{PDW1}$ is connected to AD21, and $\overline{PDW0}$ is connected to $\overline{AD20}$.

Assertion of $\overline{PDW1}$ low signals a power-down interrupt to the DSP.

Deassertion of $\overline{PDW1}$ high causes a wake-up of the DSP. The PME_EN output from the Combo Master should reflect the current PCI function PME_EN bit and should be connected to the ADSP-2192 AD20 pin. The PMI_EN bit should be set to enable interrupt and wake-up of the DSP upon any change of the PME_EN state. If PME_EN is turned off, the DSPs can wake up if necessary and then power themselves and the ADSP-2192 completely down (clocks stopped).

Table 24. Sub-ISA Indirect Access Registers

ISAA[3:1]	Name	Reset	Comments
0x0	Control Register Address	0x0000	Address and direction control for registers accesses
0x1	Reserved		
0x2	Control Register Data	0x0000	Data for register accesses
0x3	Reserved		
0x5-0x4	DSP Memory Address	0x000000	Address and direction control for DSP memory accesses
0x7-0x6	DSP Memory Data	0x000000	Data for DSP memory accesses.

PCI Interface to DSP Memory

The PCI interface can directly access the DSP memory space using DMA transfers. The transactions can be either slave transfers, in which the host initiates the transaction, or master transfers, in which the ADSP-2192 initiates the PCI transaction. The registers that control PCI DMA transfers are accessible from both the DSP (on the Peripheral Device Control Bus) and the PCI Bus.

The PCI/Sub-ISA Bus uses the Peripheral Device Control Register Space which is distributed throughout the ADSP-2192 and connected through the Peripheral Device Control Bus. The PCI bus can access these registers directly.

USB Interface to DSP Memory

The USB interface can directly access the DSP memory space using DMA transfers to memory locations specified by the USB endpoints. The registers that control USB endpoint DMA transfers are accessible from both the DSP (on the Peripheral Device Control Bus) and the USB Bus.

The Peripheral Device Control Register Space is distributed throughout the ADSP-2192 and connected through the Peripheral Device Control Bus. The USB Bus can access these registers directly.

AC'97 Codec Interface to DSP Memory

Transfers from AC'97 data to DSP memory are accomplished using DMA transfer through the DSP FIFOs. Each DSP has four FIFOs available for data transfers to/from the AC'97 Codec Interface. The registers that control FIFO DMA transfers are only accessible from within the DSP and are defined as part of the core register space.

Data FIFO Architecture

Each DSP core within the ADSP-2192 contains four FIFOs which provide a data communication path to the rest of the chip. Two of the FIFOs are input FIFOs, receiving data into the DSP. The other two FIFOs are transmit FIFOs, sending data from the DSP to the codec, AC'97 interface, or the other DSP. Each FIFO is eight words deep and sixteen bits wide. Interrupts to the DSP can be generated when some words have been received in the input FIFOs, or when some words are empty in the Transmit FIFOs.

The interface to the FIFOs on the DSP is simply a register interface to the Peripheral Interface bus. TX0, RX0, TX1, and RX1 are the primary FIFO registers in the universal register map of the DSP. The FIFOs can be used to generate interrupts to the DSP based upon FIFO transactions or can initiate DMA requests.

When communicating with the AC'97 interface, the Connection Enable bits in the control register are set to '10'. Bit 3 selects stereo or mono transfers to and from the AC'97 interface. Bits 7-4 select the AC'97 slot associated with this FIFO.

When stereo is selected, the slot identified and the next slot are both associated with the FIFO. Typically, stereo is selected for left and right data, and both left and right must be associated with the same external AC'97 codec and have their sample rates locked together. In this case, left and right data will alternate in the FIFO with the left data coming first.

If the FIFO is enabled for the AC'97 interface, and a valid request for data comes along that the FIFO cannot fulfill, the transmitter underflow bit is set, indicating that an invalid value was sent over the selected slot. Similarly, on the receive side, if the FIFO is full and another valid word is received, the Overflow bit is sent to indicate the loss of data.

FIFO Control Registers

The Transmit FIFO Control Register has the following bit field definitions:

- CE (Bits 1–0): Connection Enable (00 = Disable, 01 = Reserved, 10 = Connect to AC'97, and 11 = Reserved)
- DPSel (Bit 2): Reserved (0)
- SMSel (Bit 3): Stereo / Mono Select - AC'97 Mode Only (0 = Mono Stream or 1 = Stereo Stream)
- SLOT (Bits 7–4): AC'97 Slot Select - AC'97 Mode Only

Table 25. AC'97 Slot Select Values

Slot	Mono	Stereo
0000–0010	Reserved	
0011	Slot 3	Slots 3/4
0100	Slot 4	Slots 4/5
0101	Slot 5	Slots 5/6
0110	Slot 6	Slots 6/7
0111	Slot 7	Slots 7/8
1000	Slot 8	Slots 8/9
1001	Slot 9	Slots 9/10
1010	Slot 10	Slots 10/11
1011	Slot 11	Slots 11/12
1100	Slot 12	Not Allowed
1101–1111	Reserved	

- FIP (Bits 10–8): FIFO interrupt position. An interrupt is generated when FIP[2:0] Words remain in the FIFO. The interrupt is level-sensitive.
- DME (Bit 11): DMA Enable. (0 = DMA Disabled or 1 = DMA Enabled)
- TFF (Bit 13): Transmit FIFO Full - Read Only. (0 = FIFO Not Full or 1 = FIFO Full)
- TFE (Bit 14): Transmit FIFO Empty - Read Only. (0 = FIFO Not Empty or 1 = FIFO Empty)
- TU (Bit 15): Transmit Underflow - Sticky, Write "1" Clear. (0 = FIFO Underflow has not occurred or 1 = FIFO Underflow has occurred)

The Receive FIFO Control Register has the following bit field definitions:

- CE (Bits 1–0): Connection Enable. (00 = Disable, 01 = Reserved, 10 = Connect to AC'97, 11 = Reserved)
- DPSel (Bit 2): Reserved (0)

- SMSel (Bit 3): Stereo / Mono Select - AC'97 Mode Only. (0 = Mono Stream or 1 = Stereo Stream)
- SLOT (Bits 7–4): AC'97 Slot Select - AC'97 Mode Only.

Table 26. AC'97 Slot Select Values

Slot	Mono	Stereo
0000–0010	Reserved	
0011	Slot 3	Slots 3/4
0100	Slot 4	Slots 4/5
0101	Slot 5	Slots 5/6
0110	Slot 6	Slots 6/7
0111	Slot 7	Slots 7/8
1000	Slot 8	Slots 8/9
1001	Slot 9	Slots 9/10
1010	Slot 10	Slots 10/11
1011	Slot 11	Slots 11/12
1100	Slot 12	Not Allowed
1101–1111	Reserved	

- FIP (Bit 10–8): FIFO interrupt position. An interrupt is generated when FIP[2:0] + 1 words have been received in the FIFO. The interrupt is level-sensitive.
- DME (Bit 11): DMA Enable. (0 = DMA Disabled or 1 = DMA Enabled)
- RFF (Bit 13): Receive FIFO Full - Read Only. (0 = FIFO Not Full or 1 = FIFO Full)
- RFE (Bit 14): Receive FIFO Empty - Read Only. (0 = FIFO Not Empty or 1 = FIFO Empty)
- RO (Bit 15): Receive Overflow - Sticky, Write-One-Clear. (0 = FIFO Overflow has not occurred or 1 = FIFO Overflow has occurred)

System Reset Description

There are several sources of reset to the ADSP-2192.

- Power On Reset
- PCI Reset
- USB Reset
- Soft Reset (RST in CMSR Register)

Power On Reset

The DSP has an internal power on reset circuit that resets the DSP when power is applied. The DSP also has a Power On Reset $\overline{\text{PORST}}$ signal that can initiate this master reset.

Note that $\overline{\text{PORST}}$ is not needed when using PCI or USB (and is shown as a no connect in Figure 8 on page 33); these interfaces reset the DSP under their control as needed.

DSP Software Reset

The DSP can generate a software reset using the RSTD bit in DSP Interrupt/Powerdown Registers). Generally, reset conditions are handled by forcing the DSPs to execute ROM- or RAM-based Reset Handler code. The Reset Handler that gets executed can be dictated by the Reset Source as defined by the CRST[1:0] bits in the Chip Mode/Status Register (CMSR).

The exact Reset Functionality is therefore defined by the ROM and RAM Reset Handler Code and as such is programmable.

Booting Modes

The ADSP-2192 has two mechanisms for automatically loading internal program memory after reset. The CRST pins, sampled during power on reset, implement these modes:

- Boot from PCI Host
- Boot from USB Host

Optionally, extra boot information can come from an SPI or Microwire serial EPROM during PCI or USB booting. The boot process flow appears in Figure 6 on page 32.

Power Management Description

The ADSP-2192 supports several states with distinct power management and functionality capabilities. These states encompass both hardware and software state.

The driver and DSP code take responsibility for detailed power management of the modem, so minimum power levels are achieved regardless of OS or BIOS. The driver and DSPs manage power by changing platform states as necessary in response to events.

Power Regulators

The ADSP-2192 is intended to operate in a variety of different systems. These include PCI, CardBus, USB and imbedded (Sub-ISA) applications. The PCI and USB specifications define power consumption limits that constrain the ADSP-2192 design.

2.5V Regulator Options

In 5V and 3.3V PCI applications the ADSP-2192 2.5V IVDD supply will be generated by an on-chip regulator. The internal 2.5V supply (IVDD) can be generated by the on-chip regulator combined with an external power transistor as shown in Figure 7 on page 32. To support the PCI specification's power down modes, the two transistors control the primary and auxiliary supply. If the reference voltage on RVDD (typically the same as PCIVDD) drops out, the VCTRLAUX will switch on the device connected

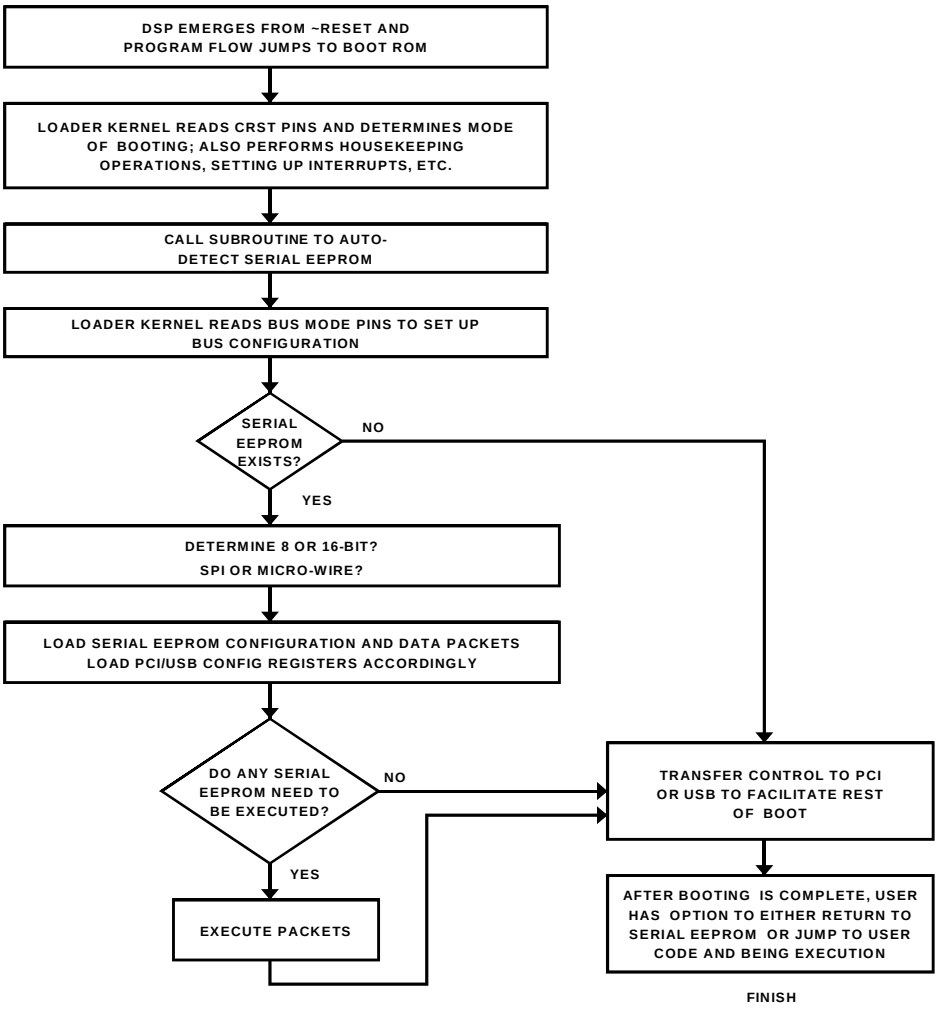

Figure 6. ADSP-2192 Boot Process Flow

to PCIVAUX and VCTRLVDD will switch off the primary supply. USB applications may require an external high-efficiency switching regulator to generate the 2.5V supply for the ADSP-2192.

Low Power Operation

In addition to supporting the PCI and USB standard's power down modes, the ADSP-2192 supports additional power down modes for the DSP core's and peripheral buses. The power down modes are controlled by the DSP1 and DSP2 Interrupt/Powerdown registers.

Clock Signals

The ADSP-2192 can be clocked by a crystal oscillator. If a crystal oscillator is used, the crystal should be connected across the XTALI/O pins, with two capacitors connected as shown in Figure 8 on page 33. Capacitor values are dependent on crystal type and should be specified by the crystal

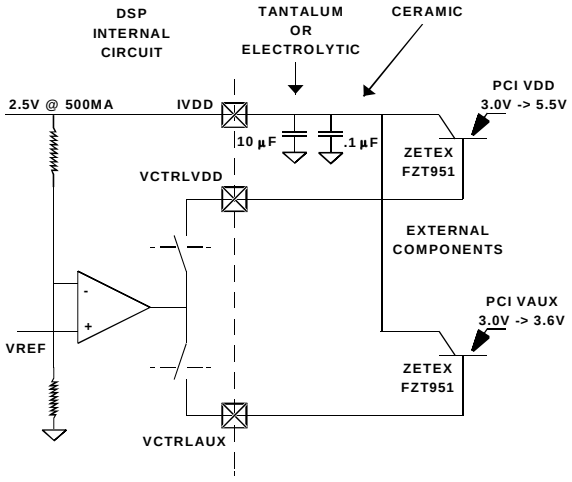

Figure 7. ADSP-2192 2.5V Regulator Options

manufacturer. A parallel-resonant, fundamental frequency, microprocessor-grade 24.576 MHz crystal should be used for this configuration.

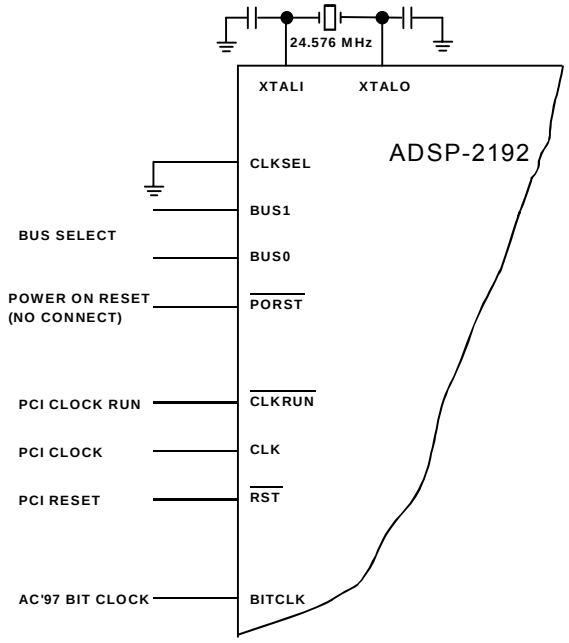

Figure 8. ADSP-2192 External Crystal Connections

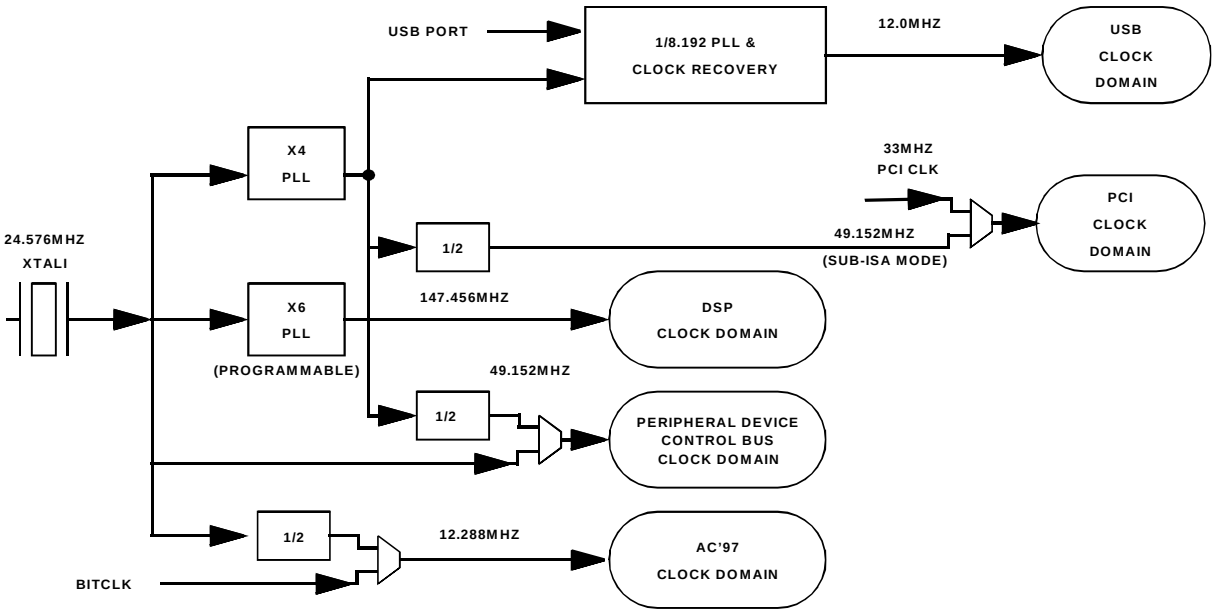

Figure 9. ADSP-2192 Clock Domains

Instruction Set Description

The ADSP-2192 assembly language instruction set has an algebraic syntax that was designed for ease of coding and readability. The assembly language, which takes full advantage of the processor's unique architecture, offers the following benefits:

- ADSP-219x assembly language syntax is a superset of and source code-compatible (except for two data registers and DAG base address registers) with ADSP-218x family syntax. You may need to restructure your 218x programs, however, to accommodate the ADSP-2192's unified memory space and to conform to its interrupt vector map.
- The algebraic syntax eliminates the need to remember cryptic assembler mnemonics. For example, a typical arithmetic add instruction, such as AR = AX0 + AY0, resembles a simple equation.
- Every instruction, except two, assembles into a single, 24-bit word that can execute in a single instruction cycle. The exceptions are two dual-word instructions, one of which writes 16- or 24-bit immediate data to memory, and the other of which jumps/calls to other pages in memory.
- Multi-function instructions allow parallel execution of an arithmetic, MAC, or shift instruction with up to two fetches or one write to processor memory space during a single instruction cycle.
- Supports a wider variety of conditional and unconditional jumps and calls and a larger set of conditions on which to base execution of conditional instructions.

Development Tools

The ADSP-2192 is supported with a complete set of VisualDSP++™ software and hardware development tools, which include Analog Devices VisualDSP++ integrated development environment, evaluation kit, and emulators. The JTAG emulator hardware used for other ADSP-219x DSPs, also fully emulates the ADSP-2192.

Both the ADSP-219x hardware development tools family and the VisualDSP++ integrated project management and debugging environment support the ADSP-2192. The VisualDSP++ project management environment enables you to develop and debug an application.

The ADSP-219x software development environment, VisualDSP++, includes an easy-to-use assembler that is based on an algebraic syntax; an archiver (librarian/library builder); a linker; a loader; a cycle-accurate, instruction-level simulator; a C/C++ compiler; and a C/C++ run-time library that includes DSP and mathematical functions. Two key points for these tools are:

- Compiled ADSP-219x C/C++ code efficiency—The compiler has been developed for efficient translation of C/C++ code to ADSP-219x assembly. The DSP has architectural features that improve the efficiency of compiled C/C++ code.
- ADSP-218x family code compatibility—The assembler has legacy features to ease the conversion of existing ADSP-218x applications to the ADSP-219x.

Debugging both C/C++ and assembly programs with the VisualDSP++ debugger, you can:

- View mixed C/C++ and assembly code (interleaved source and object information)
- Insert break points
- Set conditional breakpoints on registers, memory, and stacks
- Trace instruction execution
- Profile program execution
- Fill and dump memory
- Source level debugging
- Create custom debugger windows

The VisualDSP++ IDE lets you define and manage DSP software development. Its dialog boxes and property pages enable you to configure and manage all of the ADSP-219x development tools, including the syntax highlighting in the VisualDSP++ editor. This capability lets you:

- Control how the development tools process inputs and generate outputs.
- Maintain a one-to-one correspondence with the tool's command line switches.

Analog Devices DSP emulators use the IEEE 1149.1 JTAG test access port of the ADSP-2192 processor to monitor and control the target board processor during emulation. The emulator provides full-speed emulation, allowing inspection and modification of memory, registers, and processor stacks. Non-intrusive in-circuit emulation is assured by the use of the processor's JTAG interface; the emulator does not affect target system loading or timing.

Note that the ADSP-2192 JTAG port does not support boundary scan.

In addition to the software and hardware development tools available from Analog Devices, third parties provide a wide range of tools supporting the ADSP-219x processor family. Hardware tools include ADSP-219x PC plug-in cards. Third party software tools include DSP libraries, real-time operating systems, and block diagram design tools.

The emulator probe requires the ADSP-2192's CLKIN, TMS, TCK, $\overline{\text{TRST}}$, TDI, TDO, $\overline{\text{EMU}}$, and GND signals be made accessible on the target system via a 14-pin con-

nector (a 2 row × 7 pin strip header) such as that shown in Figure 10 on page 35. The emulator probe plugs directly onto this connector for chip-on-board emulation. You must add this connector to your target board design if you intend to use the ADSP-2192 emulator. The total trace length between the emulator connector and the furthest device sharing the emulation JTAG pins should be limited to 15 inches maximum for guaranteed operation. This length restriction must include emulation JTAG signals which are routed to one or more ADSP-2192 devices, or a combination of ADSP-2192 devices and other JTAG devices on the chain.

The 14-pin, 2-row pin strip header is keyed at the pin 3 location; pin 3 must be removed from the header. The pins must be 0.025 inch square and at least 0.20 inch in length. Pin spacing should be 0.1 × 0.1 inches. Pin strip headers are available from vendors such as 3M, McKenzie and Samtec.

The BTMS, BTCK, $\overline{\text{BTRST}}$ and BTDI signals are provided so the test access port can also be used for board-level testing. When the connector is not being used for emulation, place jumpers between the Bxxx pins and the xxx pins. If the test access port will not be used for board testing, tie $\overline{\text{BTRST}}$ and BTCK pins to GND. The $\overline{\text{TRST}}$ pin must be asserted after power-up (through $\overline{\text{BTRST}}$ on the connector) or held low for proper operation of the ADSP-2192. None of the Bxxx pins (Pins 5, 7, 9, 11) are connected on the emulator probe.

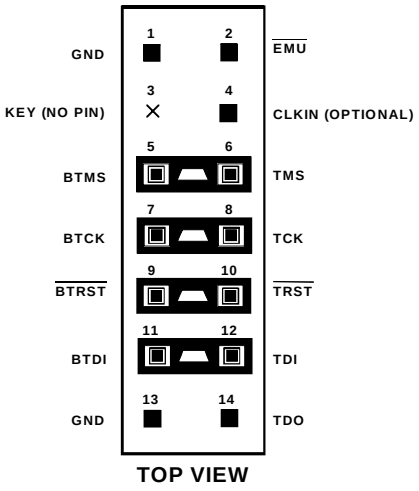

Figure 10. Target Board Connector For ADSP-2192 Analog Devices Emulator (Jumpers in Place)

The JTAG signals are terminated on the emulator probe as follows:

Table 27. Analog Devices DSP Emulator Probe Terminations

Signal	Termination
TMS	Driven through 22 Ω Resistor (16 mA Driver)
TCK	Driven at 10 MHz through 22 Ω Resistor (16 mA Driver)
$\overline{\text{TRST}}$	Active Low Driven through 22 Ω Resistor (16 mA Driver) (Pulled Up by On-Chip 20 kΩ Resistor); $\overline{\text{TRST}}$ is driven low until the emulator probe is turned on by the emulator at software start-up. After software start-up, $\overline{\text{TRST}}$ is driven high.
TDI	Driven by 22 Ω Resistor (16 mA Driver)
TDO	One TTL Load, Split (160/220)
CLKIN	One TTL Load, Split (160/220)
$\overline{\text{EMU}}$	Active Low 4.7 kΩ Pull-Up Resistor, One TTL Load (Open-Drain Output from the DSP)

Figure 11 on page 36 shows JTAG scan path connections for systems that contain multiple ADSP-2192 processors

To make it easier to evaluate the ADSP-219x DSP family for your application, Analog Devices sells the ADSP-2192 EZ-KIT Lite™. The ADSP-2192 EZ-KIT Lite provides developers with a cost-effective method for evaluating of the ADSP-219x family of DSPs. The EZ-KIT Lite includes an ADSP-2192 DSP evaluation board and fundamental debugging software. The evaluation board in this kit contains an ADSP-2192 digital signal processor, Flash Memory, Audio/Telephony type Codec, breadboard area, Flag LED, Reset/Interrupt/Flag push buttons, and ADSP-2192 peripheral port connectors. The peripheral connectors include a JTAG test and emulation port connector that supports the Analog Devices emulators and other connector locations that provide additional evaluation and interface points to the ADSP-2192 peripheral ports. The ADSP-2192 EZ-KIT Lite comes with an evaluation suite of the VisualDSP++ integrated development environment with the C/C++ compiler, assembler, and linker that supports typical debug functions including memory/register read and write, halt, run, and single step. All software tools are limited to use with the EZ-KIT Lite product.

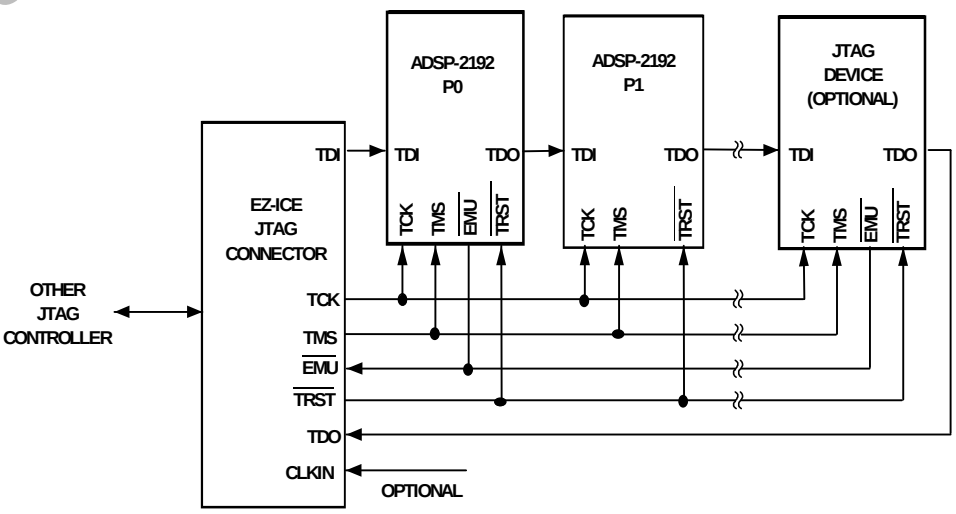

Figure 11. JTAG Scan Path Connections for Multiple ADSP-2192 Systems

Additional Information

This data sheet provides a general overview of the ADSP-2192 architecture and functionality. For detailed information on the ADSP-219x Family core architecture and instruction set, refer to the ADSP-219x/*2191 DSP Hardware Reference*.

PIN DESCRIPTIONS

ADSP-2192 pin definitions are listed in a series of tables following this section. Inputs identified as synchronous (S) must meet timing requirements with respect to CLKIN (or with respect to TCK for TMS, TDI). Inputs identified as asynchronous (A) can be asserted asynchronously to CLKIN (or to TCK for $\overline{\text{TRST}}$).

The following symbols appear in the Type columns of these tables: G = Ground, I = Input, O = Output, P = Power Supply, and T = Three-State.

Table 28. ADSP-2192 Pin Configurations: PCI/USB Bus Interface

Pin Name	LQFP	I/O	Description
AD0	57	I/O	Address and Data Bus
AD1	56	I/O	Address and Data Bus
AD2	55	I/O	Address and Data Bus
AD3	54	I/O	Address and Data Bus
AD4	53	I/O	Address and Data Bus
AD5	48	I/O	Address and Data Bus
AD6	47	I/O	Address and Data Bus
AD7	46	I/O	Address and Data Bus
AD8	44	I/O	Address and Data Bus
AD9	43	I/O	Address and Data Bus
AD10	42	I/O	Address and Data Bus
AD11	37	I/O	Address and Data Bus

Table 28. ADSP-2192 Pin Configurations: PCI/USB Bus Interface (Continued)

Pin Name	LQFP	I/O	Description
AD12	36	I/O	Address and Data Bus
AD13	35	I/O	Address and Data Bus
AD14	34	I/O	Address and Data Bus
AD15	33	I/O	Address and Data Bus
AD16	15	I/O	Address and Data Bus
AD17	14	I/O	Address and Data Bus
AD18	13	I/O	Address and Data Bus
AD19	12	I/O	Address and Data Bus
AD20	11	I/O	Address and Data Bus
AD21	8	I/O	Address and Data Bus
AD22	7	I/O	Address and Data Bus
AD23	6	I/O	Address and Data Bus
AD24	3	I/O	Address and Data Bus
AD25	2	I/O	Address and Data Bus
AD26	143	I/O	Address and Data Bus
AD27	142	I/O	Address and Data Bus
AD28	141	I/O	Address and Data Bus
AD29	138	I/O	Address and Data Bus
AD30	137	I/O	Address and Data Bus
AD31	136	I/O	Address and Data Bus
$\overline{\text{CBE0}}$	45	I/O	PCI Command / Byte Enable
$\overline{\text{CBE1}}$	32	I/O	PCI Command / Byte Enable
$\overline{\text{CBE2}}$	16	I/O	PCI Command / Byte Enable
$\overline{\text{CBE3}}$	4	I/O	PCI Command / Byte Enable
CLK	130	I	PCI Clock
$\overline{\text{CLKRUN}}$	26	O	Clock Run
$\overline{\text{DEVSEL}}$	24	I/O	PCI Target Device Select
$\overline{\text{FRAME}}$	17	I/O	PCI Frame Select
$\overline{\text{GNT}}$	131	I	Grant
IDSEL	5	I	PCI Initiator Device Select
INTAB	128	O	PCI / ISA Interrupt

Table 28. ADSP-2192 Pin Configurations: PCI/USB Bus Interface (Continued)

Pin Name	LQFP	I/O	Description
$\overline{\text{IRDY}}$	22	I/O	PCI Initiator Ready
PAR	31	I/O	PCI Bus Parity/
PCIGND	1, 10, 21, 30, 39, 52, 133	I	PCI Ground
PCIVDD	9, 18, 29, 38, 51, 132, 144	I	PCI Vdd supply
$\overline{\text{PERR}}$	27	I/O	PCI Parity Error/ USB- (Inverting input)
$\overline{\text{PME}}$	135	O	PCI Power Management Event
$\overline{\text{REQ}}$	134	O	Request
$\overline{\text{RST}}$	129	I	PCI Reset
$\overline{\text{SERR}}$	28	O	PCI System Error/ USB+ (Non-inverting input).-5-
$\overline{\text{STOP}}$	25	I/O	PCI Target Stop
$\overline{\text{TRDY}}$	23	I/O	PCI Target Ready

Table 29. Pin Configurations: Crystal / Configuration Pins

Pin Name	LQFP	I/O	Description
BUS0	124	I	PCI/ Sub-ISA /CardBus Select pins
BUS1	123	I	PCI/ Sub-ISA /CardBus Select pins
CLKSEL	116	I/O	Clock Select
IGND	122	I	IGND
NC	127	O	No Connect
$\overline{\text{PORST}}$	121	I	Power On Reset
XTALI	118	I	Crystal input pin (24.576MHz)
XTALO	119	I/O	Crystal output pin

Table 30. Pin Configurations: Analog Pins

Pin Name	LQFP	I/O	Description
AGND	67	I	Analog Ground
AQGND	68	I	Reference Analog Ground
CTRLAUX	61	I	X supply

Table 30. Pin Configurations: Analog Pins (Continued)

Pin Name	LQFP	I/O	Description
CTRLVDD	63	I	Control Vdd
IVDD	62	I	Digital Vdd
NC	66	O	No Connect
NC	69	I	No Connect
NC	70	I	No Connect
RVAUX	60	I	X supply
RVDD	64	I	Analog Vdd supply

Table 31. Pin Configurations: AC'97 Interface Pins

Pin Name	LQFP	I/O	Description
$\overline{\text{ACRST}}$	102	O	AC'97 Reset
ACVAUX	92	I	AC'97 Vaux input
ACVDD	93	I	AC'97 Vdd input
BITCLK	96	I	AC'97 Bit Clock
SDI0	99	I	AC'97 Serial Data Input, bit 0
SDI1	98	I	AC'97 Serial Data Input, bit 1
SDI2	97	I	AC'97 Serial Data Input, bit 2
SDO	100	O	AC'97 Serial Data Output
SYNC	101	O	AC'97 Sync

Table 32. Pin Configurations: Serial EEPROM Pins

Pin Name	LQFP	I/O	Description
SCK	72	I	Serial EEPROM Clock
SDA	71	I	Serial EEPROM Data
SEN	73	I	Serial EEPROM Enable

Table 33. Pin Configurations: Emulator Pins

Pin Name	LQFP	I/O	Description
$\overline{\text{EMU}}$	74	O	Emulator Event Pin
TCK	78	I	Emulator Clock Input
TDI	80	I	Emulator Data Input

Table 33. Pin Configurations: Emulator Pins (Continued)

Pin Name	LQFP	I/O	Description
TDO	81	O	Emulator Data Output
TMS	75	I	Emulator Mode Select
$\overline{\text{TRST}}$	79	I	Emulator Logic Reset

Table 34. Pin Configurations: GPIO Pins

Pin Name	LQFP	I/O	Description
AIOGND	76, 91		GPIO Ground
IO0	82	I/O	GPIO Pin, bit 0
IO1	83	I/O	GPIO Pin, bit 1
IO2	84	I/O	GPIO Pin, bit 2
IO3	86	I/O	GPIO Pin, bit 3
IO4	87	I/O	GPIO Pin, bit 4
IO5	88	I/O	GPIO Pin, bit 5
IO6	89	I/O	GPIO Pin, bit 6
IO7	90	I/O	GPIO Pin, bit 7
IOVDD	77, 85		GPIO Vdd

Table 35. Pin Configurations: Reserved Pins

Pin Name	LQFP	I/O	Description
ACVAUX	113	I	ACVAUX Supply
ACVDD	112	I	AC Vdd
GND	111	I	Ground
NC	94	O	No Connect
NC	105	O	No Connect
NC	106	I	No Connect
NC	107	I	No Connect
NC	108		No Connect
NC	109	I	No Connect
NC	110	I	No Connect
NC	114	I	No Connect
NC	115	I	No Connect
NC	95	O	No Connect

Table 36. Pin Configurations: Power Supply Pins

Pin Name	LQFP	I/O	Description
ACVAUX	92		
AIOGND	91		
AVDD	65		Analog VDD supply
CTRLAUX	61		
CTRLVDD	63		
IGND	20, 41, 50, 59, 104, 120, 126, 139		Digital Ground
IGND	122		
IVDD	19, 40, 49, 58, 103, 117, 125, 140		Digital Vdd
IVDD	62		
RVAUX	60		
RVDD	64		

PRELIMINARY TECHNICAL DATA

ADSP-2192—SPECIFICATIONS

Note that component specifications are subject to change without notice.

RECOMMENDED OPERATING CONDITIONS

Signal	K Grade Parameter	Min	Max	Units
V_{DDINT}[1,2]	Internal Supply Voltage	2.38	2.62	V
V_{DDEXT}[3]	External Supply Voltage Option 3.3V (All Supplies)	3.13	3.47	V
V_{DDEXT}[4]	External Supply Voltage Option 5.0V (VDD Supplies only)	4.75	5.25	V
V_{IH1}	High Level Input Voltage[5], @ V_{DDEXT} = max	2.0	V_{DDEXT}	V
V_{IH2}	High Level Input Voltage[6], @ V_{DDEXT} = max	2.2	V_{DDEXT}	V
V_{IL}	Low Level Input Voltage[1, 2], @ V_{DDEXT} = min	–0.3	0.6	V
T_{AMB}	Ambient Operating Temperature	0	+70	°C

[1] V_{DDINT} = IVDD.

[2] The "Recommended Operating Conditions" on page 42 identify V_{DDINT} as input to the ADSP-2192.

[3] V_{DDEXT} = IOVDD, PCIVDD, ACVDD, RVDD, RVAUX, ACVAUX.

[4] V_{DDEXT} = IOVDD, PCIVDD, ACVDD, RVDD only.

[5] Applies to input and bidirectional pins.

[6] Applies to input pins.

ELECTRICAL CHARACTERISTICS

Parameter		Test Conditions	Min	Max	Units
V_{OH}	High Level Output Voltage[1]	@ V_{DDEXT} = min, I_{OH} = –0.5 mA	2.4		V
V_{OL}	Low Level Output Voltage[1]	@ V_{DDEXT} = min, I_{OL} = 2.0 mA		0.4	V
I_{IH}	High Level Input Current[2, 3]	@ V_{DDEXT} = max, V_{IN} = V_{DD} max		TBD	µA
I_{IL}	Low Level Input Current[2]	@ V_{DDEXT} = max, V_{IN} = 0 V		TBD	µA
I_{ILP}	Low Level Input Current[3]	@ V_{DDEXT} = max, V_{IN} = 0 V		TBD	µA
I_{OZH}	Three-State Leakage Current[4, 5]	@ V_{DDEXT}= max, V_{IN} = V_{DD} max		TBD	µA
I_{OZL}	Three-State Leakage Current[4]	@ V_{DDEXT} = max, V_{IN} = 0 V		TBD	µA

ELECTRICAL CHARACTERISTICS (CONTINUED)

Parameter		Test Conditions	Min	Max	Units
I_{DD}	Supply Current Dynamic (Internal)	@ 160 MIPS		TBD	mA
$I_{DD-IDLE}$	Supply Current (Idle)	V_{DDINT} = 2.5V		TBD	mA
C_{IN}	Input Capacitance[6, 7]	f_{IN}=1 MHz, T_{CASE}=25°C, V_{IN}=2.5V		TBD	pF

[1] Applies to output and bidirectional pins.

[2] Applies to input.

[3] Applies to input pins with internal pull-ups.

[4] Applies to three-statable pins.

[5] Applies to three-statable pins with internal pull-ups.

[6] Applies to all signal pins.

[7] Guaranteed, but not tested.

ABSOLUTE MAXIMUM RATINGS

Parameter	Min	Max	Units
Digital Power Supply, External (V_{DDEXT})	–0.3	6.0	V
Analog Power Supply (VCC)	–0.3	6.0	V
Input Current (except supply pins)	±10.0		mA
Analog Input Voltage (Signal Pins)	–0.3	VCC +0.3	V
Digital Input Voltage (Signal Pins)	–0.3	VDD +0.3	V
Ambient Temperature (Operating)	0	+70	°C
Storage Temperature	–65	+150	°C

ESD SENSITIVITY

CAUTION: ESD (electrostatic discharge) sensitive device. Electrostatic charges as high as 4000V readily accumulate on the human body and test equipment and can discharge without detection. Although the ADSP-2192 features proprietary ESD protection circuitry, permanent damage may occur on devices subjected to high energy electrostatic discharges. Therefore, proper ESD precautions are recommended to avoid performance degradation or loss of functionality.

TIMING SPECIFICATIONS

Table 37. Sub-ISA Interface Timing Parameters

Parameters	Description	Min.	Typ	Max	Units
t_{STW}	IOR / IOW Strobe Width	100			ns
t_{ICYC}	IOR / IOW Cycle Time	240			ns
t_{AESU}	AEN Setup to IOR / IOW Falling	10			ns
t_{AEHD}	AEN Hold from IOR / IOW Rising	0			ns
t_{ADSU}	Address Setup to IOR / IOW Falling	10			ns
t_{ADHD}	Address Hold from IOR / IOW Rising	0			ns
t_{DHD1}	Data Hold from IOR Rising			20	ns
t_{DHD2}	Data Hold from IOW Rising	15			ns
t_{RDDV}	IOR Falling to Valid Read Data			40	ns
t_{WDSU}	Write Data Setup to IOW Rising	10			ns
t_{RDY1}	IOR / IOW Rising from IOCHRDY Rising	0			ns
t_{RDY2}	IOCHRDY Falling from IOR / IOW Rising	20			ns

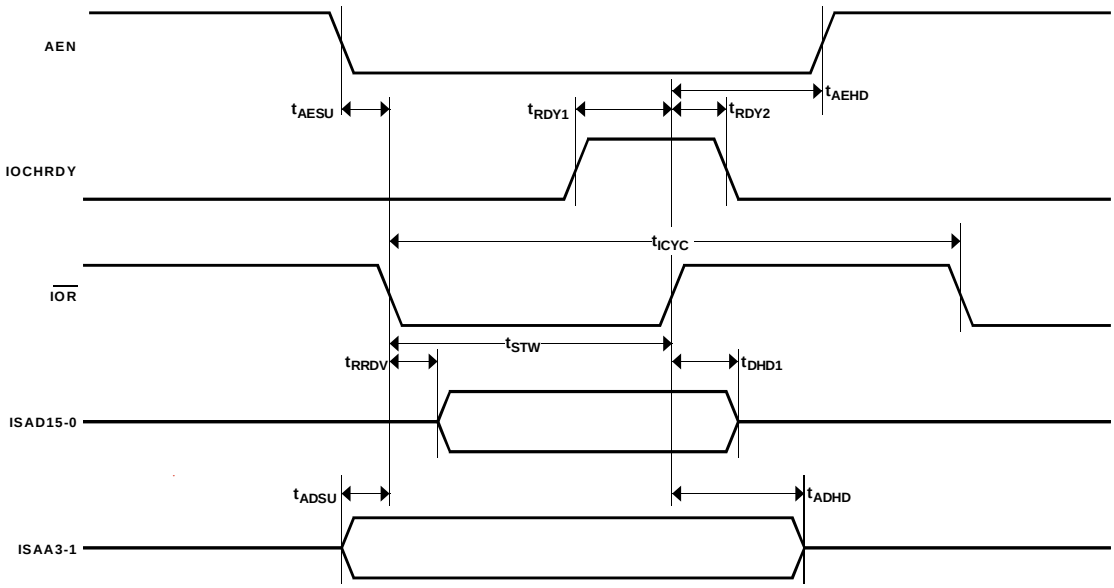

Figure 12. Sub-ISA Interface Read Cycle Timing Diagram

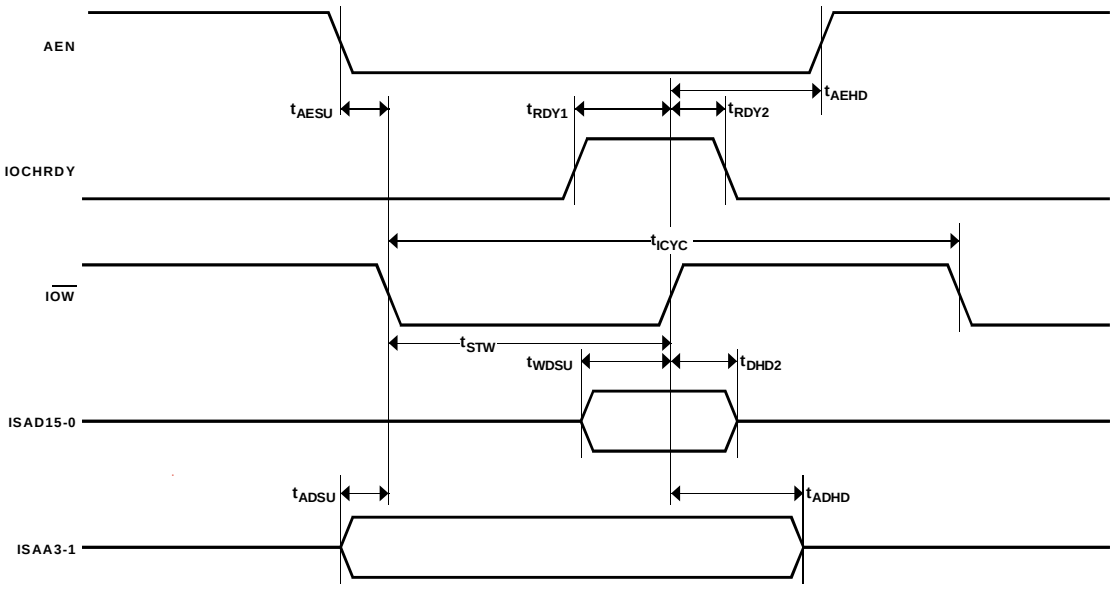

Figure 13. Sub-ISA Interface Write Cycle Timing Diagram

OUTPUT DRIVE CURRENTS

The typical I-V characteristics for the output drivers of the ADSP-2192 are TBD.

POWER DISSIPATION

Total power dissipation for the ADSP-2192 is TBD.

TEST CONDITIONS

The ADSP-2192 is tested for compliance with all support industry standard interfaces (PCI, USB, and AC'97). Also, the DSP is tested for output enable, disable, and hold time. These values (output enable, disable, and hold time) for the ADSP-2192 are TBD.

ENVIRONMENTAL CONDITIONS

The thermal characteristics in which the DSP is operating influence performance.

ENVIRONMENTAL CONDITIONS[1]

Rating Description	Symbol	LQFP
Thermal Resistance (Case-to-Ambient)	θ_{CA}	TBD °C/W
Thermal Resistance (Junction-to-Ambient)	θ_{JA}	TBD °C/W
Thermal Resistance (Junction-to-Case)	θ_{JC}	TBD °C/W

[1] Where the Ambient Temperature Rating (T_{AMB}) is:
$T_{AMB} = T_{CASE} - (PD \times \theta_{CA})$
T_{CASE} = Case Temperature in °C
PD = Power Dissipation in W

ADSP-2192 144-LEAD LQFP PINOUT

Table 38 lists the LQFP pinout by signal.

Table 38. 144-Lead LQFP Pins By Signal

SIGNAL	PIN #
$\overline{\text{ACRST}}$	102
ACVAUX	92
ACVDD	93
AD0	57
AD1	56
AD2	55
AD3	54
AD4	53
AD5	48
AD6	47
AD7	46
AD8	44
AD9	43
AD10	42
AD11	37
AD12	36
AD13	35
AD14	34
AD15	33
AD16	15
AD17	14
AD18	13
AD19	12

Table 38. 144-Lead LQFP Pins By Signal

SIGNAL	PIN #
AD20	11
AD21	8
AD22	7
AD23	6
AD24	3
AD25	2
AD26	143
AD27	142
AD28	141
AD29	138
AD30	137
AD31	136
AGND	67
AIOGND	91
AQGND	68
AVDD	65
ACVAUX	113
ACVDD	112
BITCLK	96
BUS0	124
BUS1	123
$\overline{\text{CBE0}}$	45
$\overline{\text{CBE1}}$	32
$\overline{\text{CBE2}}$	16
$\overline{\text{CBE3}}$	4
CLK	130
$\overline{\text{CLKRUN}}$	26
CLKSEL	116
CTRLAUX	61
CTRLVDD	63

Table 38. 144-Lead LQFP Pins By Signal

SIGNAL	PIN #
$\overline{\text{DEVSEL}}$	24
EMU	74
$\overline{\text{FRAME}}$	17
GND	111
$\overline{\text{GNT}}$	131
IDSEL	5
IGND	20
IGND	41
IGND	50
IGND	59
IGND	104
IGND	120
IGND	122
IGND	126
IGND	139
INTAB	128
IO0	82
IO1	83
IO2	84
IO3	86
IO4	87
IO5	88
IO6	89
IO7	90
IOGND	76
IOVDD	77
IOVDD	85
$\overline{\text{IRDY}}$	22
IVDD	19
IVDD	40

Table 38. 144-Lead LQFP Pins By Signal

SIGNAL	PIN #
IVDD	49
IVDD	58
IVDD	103
IVDD	117
IVDD	125
IVDD	140
IVDD	62
NC	115
NC	114
NC	108
NC	105
NC	109
NC	107
NC	106
NC	110
NC	127
NC	70
NC	66
NC	94
NC	69
NC	95
PAR	31
PCIGND	1
PCIGND	10
PCIGND	21
PCIGND	30
PCIGND	39
PCIGND	52
PCIGND	133
PCIVDD	9

Table 38. 144-Lead LQFP Pins By Signal

SIGNAL	PIN #
PCIVDD	18
PCIVDD	29
PCIVDD	38
PCIVDD	51
PCIVDD	132
PCIVDD	144
$\overline{\text{PERR}}$	27
$\overline{\text{PME}}$	135
$\overline{\text{PORST}}$	121
$\overline{\text{REQ}}$	134
$\overline{\text{RST}}$	129
RVAUX	60
RVDD	64
SCK	72
SDA	71
SDI0	99
SDI1	98
SDI2	97
SDO	100
SEN	73
$\overline{\text{SERR}}$	28
$\overline{\text{STOP}}$	25
SYNC	101
TCK	78
TDI	80
TDO	81
TMS	75
$\overline{\text{TRDY}}$	23

Table 38. 144-Lead LQFP Pins By Signal

SIGNAL	PIN #
$\overline{\text{TRST}}$	79
XTALI	118
XTALO	119

Table 39 lists the LQFP pinout by pin number.

Table 39. 144-Lead LQFP Pins By Pin #

SIGNAL	PIN #
PCIGND	1
AD25	2
AD24	3
$\overline{\text{CBE3}}$	4
IDSEL	5
AD23	6
AD22	7
AD21	8
PCIVDD	9
PCIGND	10
AD20	11
AD19	12
AD18	13
AD17	14
AD16	15
$\overline{\text{CBE2}}$	16
$\overline{\text{FRAME}}$	17
PCIVDD	18
IVDD	19
IGND	20
PCIGND	21
$\overline{\text{IRDY}}$	22

Table 39. 144-Lead LQFP Pins By Pin # (Continued)

SIGNAL	PIN #
$\overline{\text{TRDY}}$	23
$\overline{\text{DEVSEL}}$	24
$\overline{\text{STOP}}$	25
$\overline{\text{CLKRUN}}$	26
$\overline{\text{PERR}}$	27
$\overline{\text{SERR}}$	28
PCIVDD	29
PCIGND	30
PAR	31
$\overline{\text{CBE1}}$	32
AD15	33
AD14	34
AD13	35
AD12	36
AD11	37
PCIVDD	38
PCIGND	39
IVDD	40
IGND	41
AD10	42
AD9	43
AD8	44
$\overline{\text{CBE0}}$	45
AD7	46
AD6	47
AD5	48
IVDD	49
IGND	50
PCIVDD	51
PCIGND	52

Table 39. 144-Lead LQFP Pins By Pin # (Continued)

SIGNAL	PIN #
AD4	53
AD3	54
AD2	55
AD1	56
AD0	57
IVDD	58
IGND	59
RVAUX	60
CTRLAUX	61
IVDD	62
CTRLVDD	63
RVDD	64
AVDD	65
NC	66
AGND	67
AQGND	68
NC	69
NC	70
SDA	71
SCK	72
SEN	73
EMU	74
TMS	75
IOGND	76
IOVDD	77
TCK	78
$\overline{\text{TRST}}$	79
TDI	80
TDO	81
IO0	82

Table 39. 144-Lead LQFP Pins By Pin # (Continued)

SIGNAL	PIN #
IO1	83
IO2	84
IOVDD	85
IO3	86
IO4	87
IO5	88
IO6	89
IO7	90
AIOGND	91
ACVAUX	92
ACVDD	93
NC	94
NC	95
BITCLK	96
SDI2	97
SDI1	98
SDI0	99
SDO	100
SYNC	101
$\overline{\text{ACRST}}$	102
IVDD	103
IGND	104
NC	105
NC	106
NC	107
NC	108
NC	109
NC	110
GND	111
ACVDD	112

Table 39. 144-Lead LQFP Pins By Pin # (Continued)

SIGNAL	PIN #
ACVAUX	113
NC	114
NC	115
CLKSEL	116
IVDD	117
XTALI	118
XTALO	119
IGND	120
$\overline{\text{PORST}}$	121
IGND	122
BUS1	123
BUS0	124
IVDD	125
IGND	126
NC	127
INTAB	128
$\overline{\text{RST}}$	129
CLK	130
$\overline{\text{GNT}}$	131
PCIVDD	132
PCIGND	133
$\overline{\text{REQ}}$	134
$\overline{\text{PME}}$	135
AD31	136
AD30	137
AD29	138
IGND	139
IVDD	140
AD28	141

Table 39. 144-Lead LQFP Pins By Pin # (Continued)

SIGNAL	PIN #
AD27	142
AD26	143
PCIVDD	144

PACKAGE DIMENSIONS

The ADSP-2192 comes in a 20 mm × 20 mm, 144-lead LQFP package. All dimensions in Figure 14 on page 50 are in millimeters (mm).

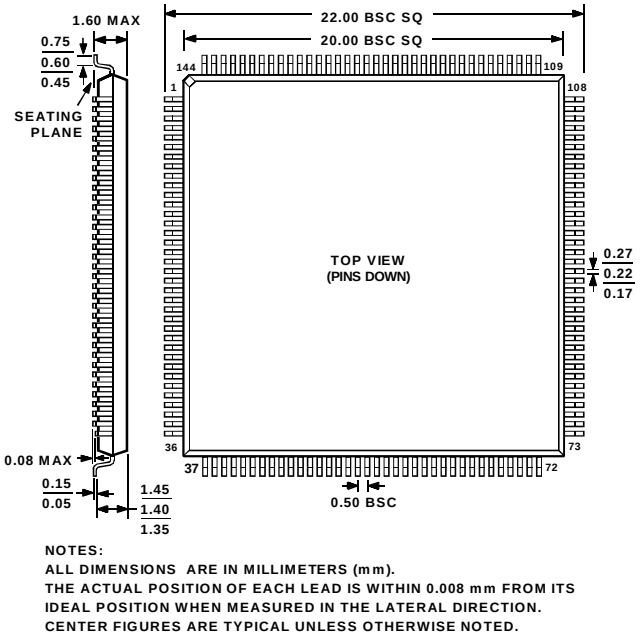

Figure 14. 20 mm x 20 mm, 144-lead LQFP Package

ORDERING GUIDE

Part Number[1]	Ambient Temperature Range	Instruction Rate	On-Chip SRAM	Operating Voltage
ADSP219212MKST160X	0°C to +70°C	160 MHz	2.4 Mbit	3 or 5 Volt

[1] ST = Plastic Thin Quad Flatpack (LQFP).